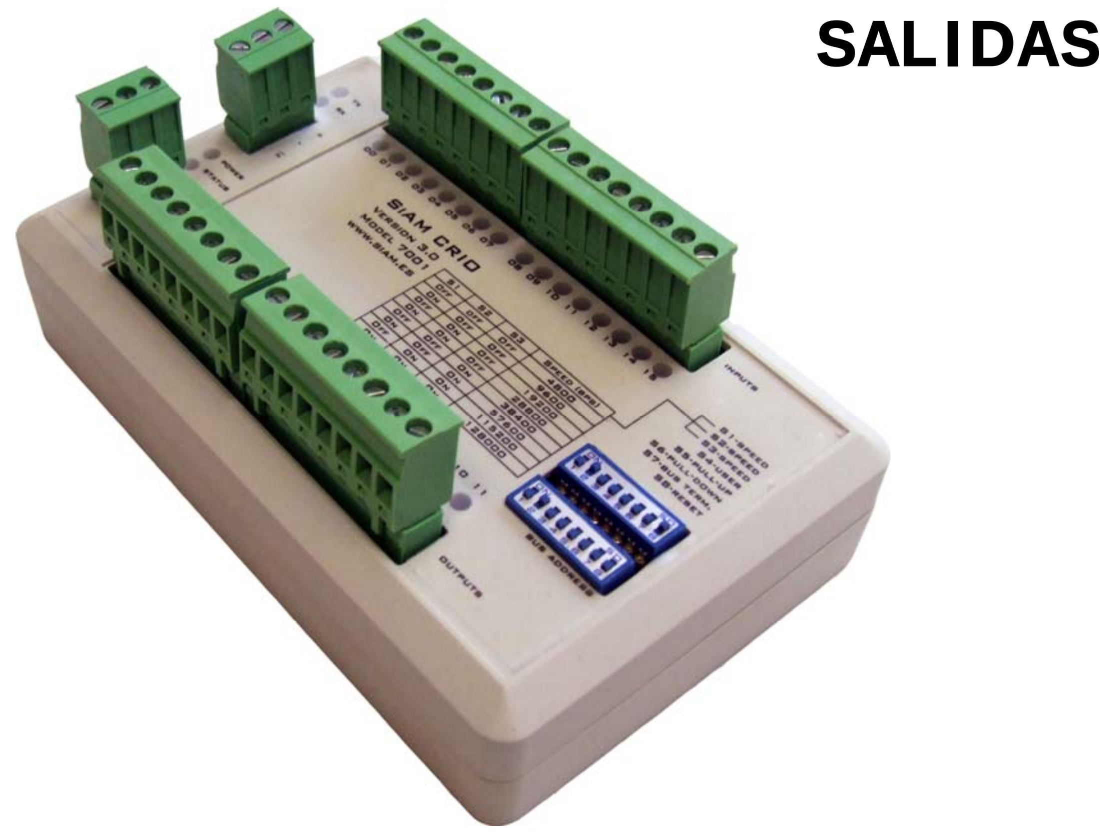


SiAM CRIO v3.0



CONTROLADOR REMOTO DE ENTRADAS Y SALIDAS



Manual del Usuario

Versión 3.0



CONTENIDOS


SIAM
SOLUCIONES TECNOLÓGICAS

1.- CARACTERÍSTICAS

1. Controlador programable con entradas/salidas integradas:
    - Programable en lenguaje de lista de instrucciones (IL) de acuerdo con IEC 1131-3.
    - 8192 bytes de programa (1300-8000 IEC instrucciones).
    - 512 palabras de usuario (256 retentivas, 256 no retentivas).
    - Soporte de los siguientes tipos de dato IEC : BOOL, BYTE, SINT, WORD, INT, DWORD, DINT, REAL.
2. Entradas/Salidas:
    - 16 entradas digitales 12/24VDC optoacopladas.
    - 12 salidas digitales 12/24VDC optoacopladas.
    - Hasta 8 entradas analógicas 0-5V o 0-20mA de 10 bits de resolución.
    - Alimentación independiente para las salidas digitales.
3. Comunicaciones:
    - RS-485 (hasta 64 dispositivos en el bus) o comunicación serie RS-232.
    - Funcionamiento a 8 velocidades diferentes: 4800 bps, 9600 bps, 19200 bps, 28800 bps, 38400 bps, 57600 bps, 115200 bps, 128000 bps.
    - Resistencia de final de bus para evitar reflexiones, pull-up y pull-down para estabilizar bus (seleccionable mediante jumpers).
    - Indicación del estado de la transmisión y recepción por medio de LEDs.
    - Protocolo Modbus RTU esclavo. Los siguientes comandos son soportados:
        - 03 (0x03) Read Holding Registers
        - 04 (0x04) Read Input Registers
        - 06 (0x06) Write Single Register
        - 16 (0x10) Write Multiple Registers
        - 17 (0x11) Report Slave ID
        - 22 (0x16) Mask Write Register
        - 23 (0x17) Read/Write Multiple Registers
        - 43/14 (0x2B/0x0D) Read Device Identification
    - Registros modbus de direccionamiento indirect para la optimización de los ciclos de comunicación.
4. Fuente de alimentación externa 9-30VDC.
5. Indicación del estado de la alimentación de la placa mediante LEDs.
6. Software gratito para la programación, configuración y diagnostico.
7. Conectores de tornillo enchufables.
8. Encapsulado para montaje en carril DIN.
9. Dimensiones: 144 mm x 90 mm x 31mm.

2.- INTRODUCCIÓN

SiAM CRIO es un controlador remoto programable de entradas y salidas con comunicación modbus como esclavo integrada. Puede ser usado como carta de entradas y salidas en un bus de campo controlado por cualquier dispositivo que implemente el protocolo modbus maestro, como pueden ser PCs, controladores programables (PLCs) o paneles de operadores (HMIs), etc.

Esta topología de bus de campo presenta un gran número de ventajas respecto a los sistemas de control centralizado, ya que permite distribuir los módulos de entradas y salidas reduciendo así el cableado necesario para sensores y actuadores, y reduciendo también el tamaño de los cuadros eléctricos. Ademas el controlador central (PC o PLC) puede ser colocado en cualquier lugar de la planta, como por ejemplo en una sala de control.

Esta nueva versión puede ademas funcionar de forma autónoma como un PLC o autómata programable, realizando pequeñas tareas de control programadas en lenguaje IL (instruction list) del estándar IEC 1131-3. Por supuesto, pueden combinarse ambos modos de funcionamiento (carta remota y PLC) y realizar así un control descentralizado.

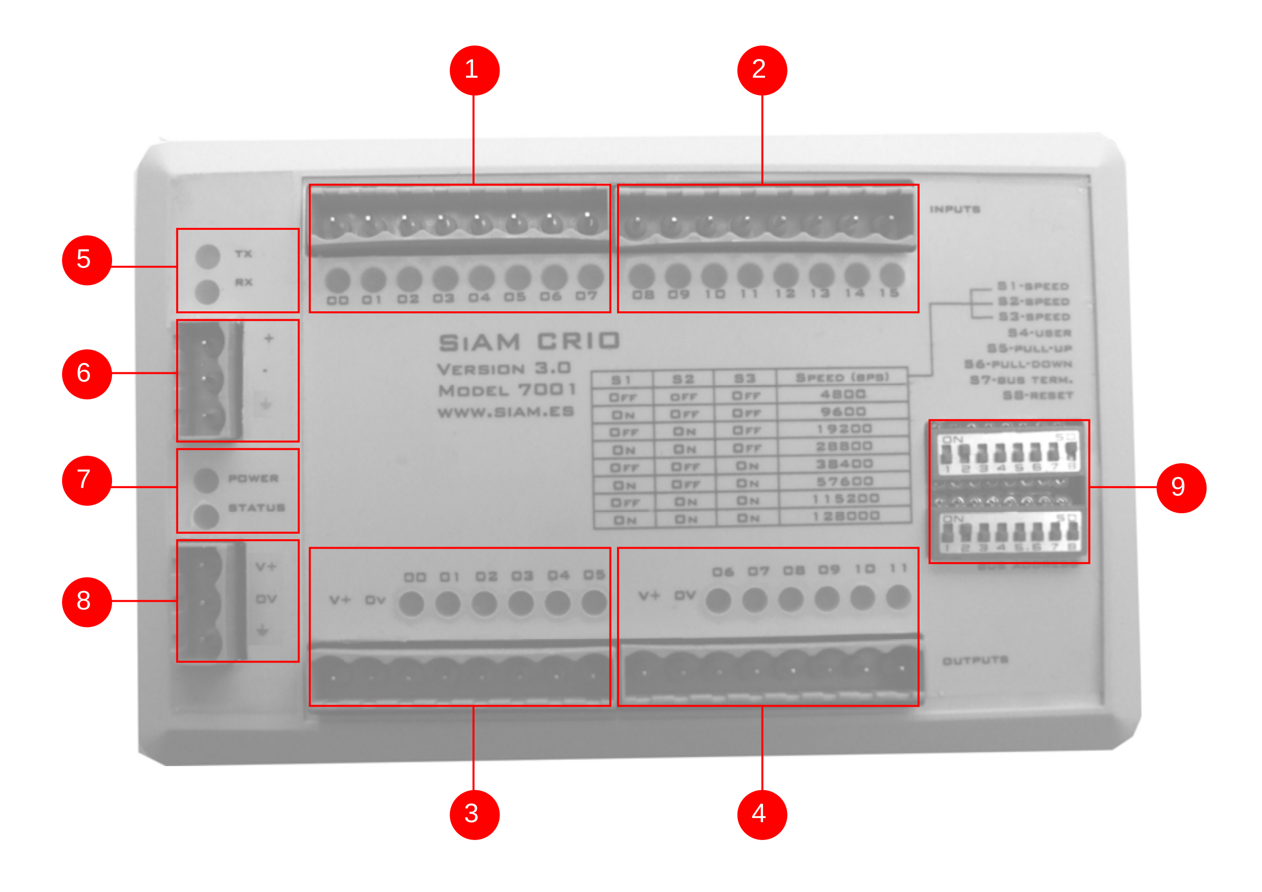
La carta del SiAM CRIO tiene 12 salidas digitales (DO), 16 entradas digitales (DI) y 8 entradas analógicas (I), las cuales son multiplexadas con las entradas digitales 8-15.

Pueden ser conectados hasta 255 módulos SiAM CRIO en una red modbus, aunque este protocolo sólo permite el direccionamiento de 247 nodos. Por tanto, en una red modbus podríamos llegar a tener hasta 2964 salidas digitales, 3952 entradas digitales y 1976 entradas analógicas.

Hay dos versiones disponibles del SiAM CRIO: una con Puerto RS-485 (3 tornillos de conexión) y otro con puerto RS-232 (conector DB9 macho).

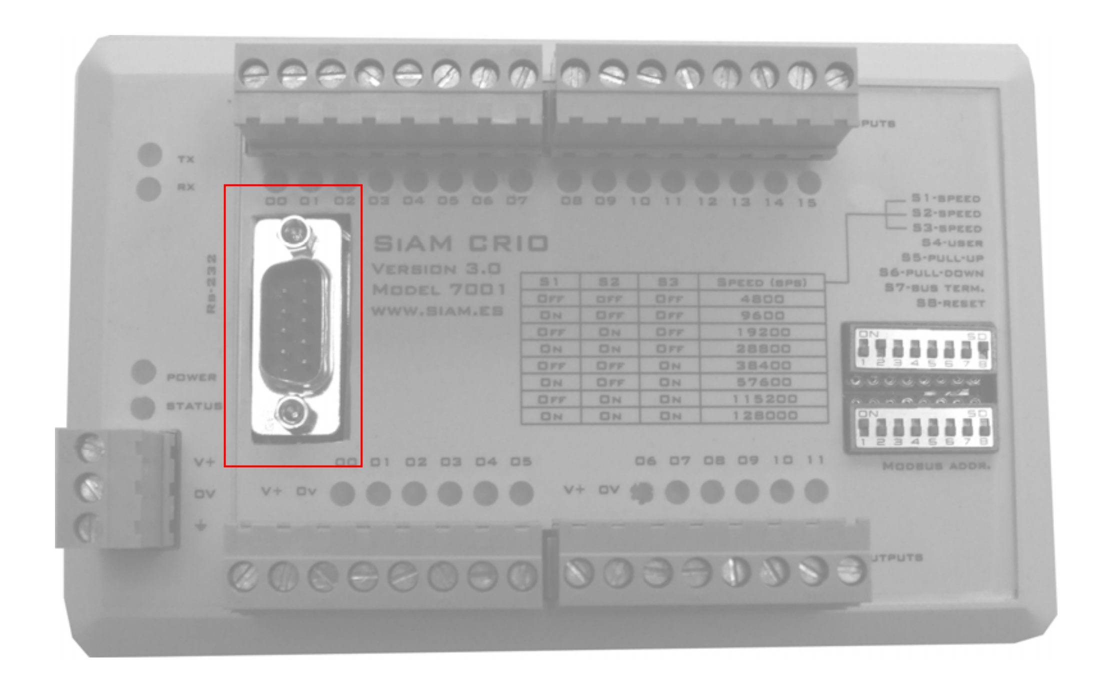
La configuración, programación y monitorización del controlador puede ser realizada con el software SiAM CRIOSoft, el cual puede ser descargado gratuitamente de la pagina web www.siam.es.

En esta imagen se muestra la versión con conector RS-485, en donde también se puede observar los LEDs y conectores:



1. Conectores y LEDs de las entradas digitales DI00…DI07
2. Conectores y LEDs de las entradas digitales DI08…DI15 y de las entradas analógicas AI0…AI7
3. Conectores y LEDs de las salidas digitales DO00…DO05
4. Conectores y LEDs de las salidas digitales DO06…DO11
5. LEDs de estado de la comunicación
6. Conector de comunicación del bus (RS-485)
7. LEDs de estado de la alimentación y del estado de la placa
8. Conector de alimentación
9. Switches de configuración

La placa con conector RS-232 tan sólo difiere, de la vista en la imagen anterior, en el conector de comunicación, como se puede ver a continuación:

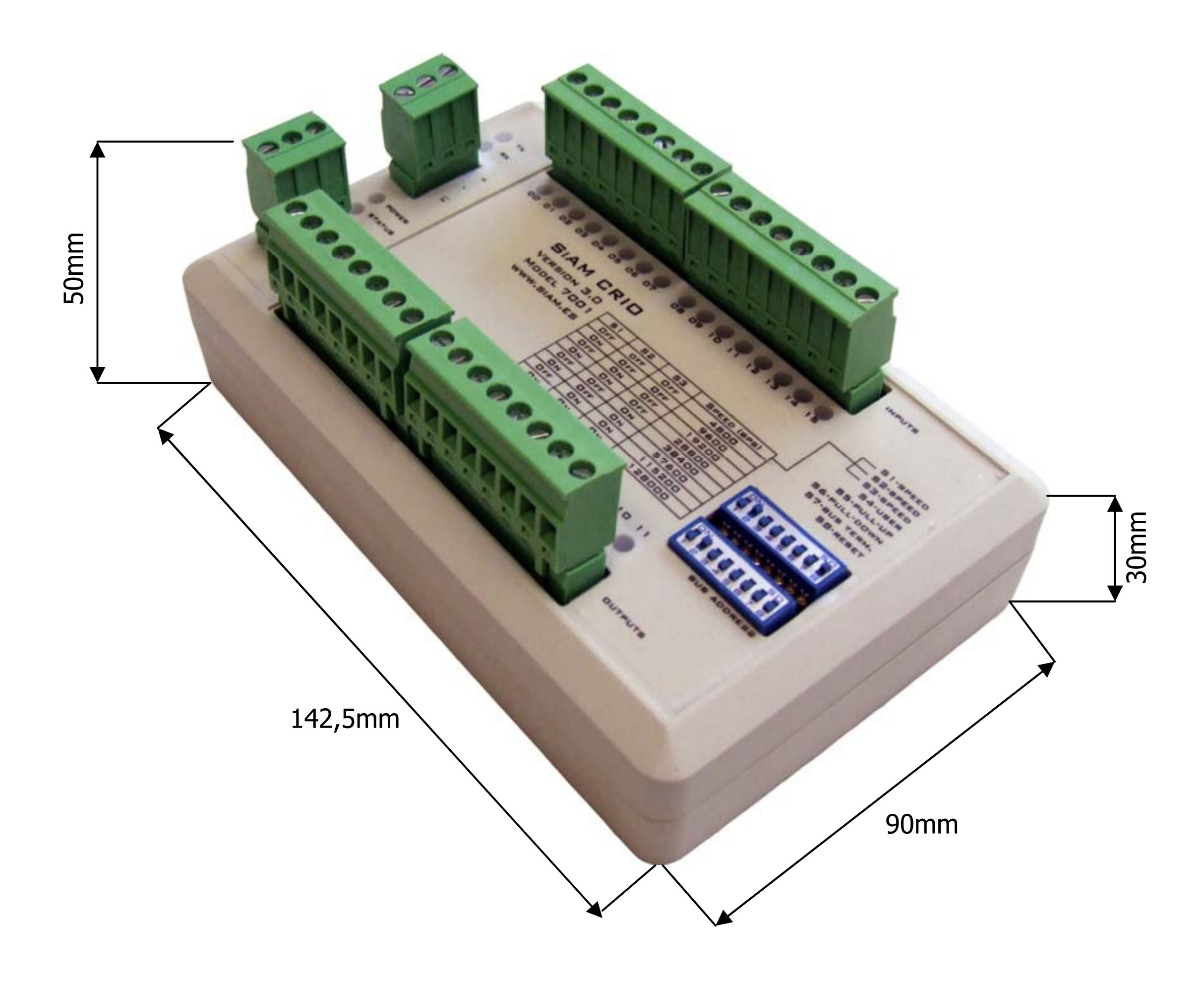


El resto de conectores y LEDs son los mismos en ambas versiones.

3.- INSTALACIÓN

El SiAM CRIO, como cualquier controlador electrónico, debe ser situado tan lejos como sea posible de las fuentes de calor, líneas de alta tensión o cualquier otra fuente de ruido eléctrico tales como motores de AC.

Las dimensiones de la placa son las siguientes:

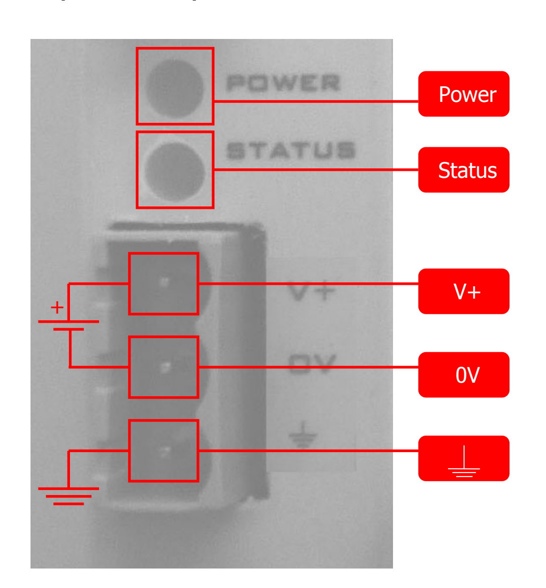


El encapsulado de la placa está equipado con dos clips para el montaje en carriles DIN, lo cuales hacen la instalación muy fácil. Para quitar la placa del carril DIN tirar con un destornillador grande para liberar los clips.

4.- CONEXIONADO

4.1.- Alimentación de la placa

La fuente de alimentación proporciona la energía necesaria a los dispositivos electrónicos de la placa para su funcionamiento. La tensión de alimentación puede ser cualquiera entre 9 y 30VDC y se conecta a través de los conectores mostrados en la figura siguiente:



Conector	Descripción
V+	Voltaje de Alimentación (9-30 VDC)
0V	Masa
⏚	Toma a tierra

El LED que muestra la palabra Power se encenderá indicando que la placa está alimentada. Si la placa tiene alimentación pero el LED no está encendido, entonces la placa tiene algún problema de hardware y necesita ser reparada.

El LED de Status indica el estado actual del dispositivo de acuerdo con la siguiente tabla:

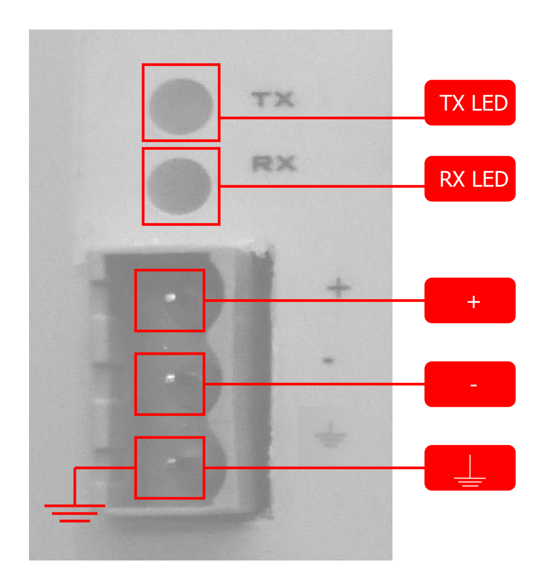
Tipo de parpadeo	Device status
1 segundo on, 1 segundo off	Modo STOP. El programa está vacío o parado por el usuario.
1 parpadeo rápido (0.1 segundos on, 0.5 segundos off)	Modo RUN. El programa se está ejecutando correctamente y sin ningún error.
2 parpadeos rápidos	Modo RUN. El programa se está ejecutando con warnings. Usar el software para determinar cuáles son.
3 parpadeos rápidos	Modo STOP. La ejecución del programa fue abortado debido a errores de programación. Usar el software para determinar cuáles son dichos errores.

Este LED siempre debe estar parpadeando. Si permanece siempre encendido o apagado entonces debe haber algún problema de hardware y la placa debe ser reparada.

4.2.- Comunicación

4.2.1.- Modulo RS-485

Esta versión usa un conector enchufable de 3 vías descrito en la siguiente figura para implementar un bus de 2 hilos:



Conector	Descripción
+	Terminal positivo RS-485, también conocido como 485+, línea A o línea D0 modbus
-	Terminal negativo RS-485, también conocido como 485-, línea B o línea D1 modbus
⏚	Pantalla de cable. Se conecta internamente a la tierra de la fuente de alimentación.

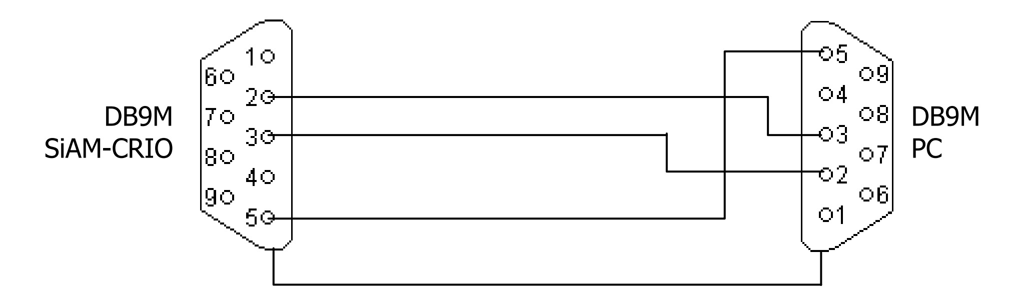
Los LEDs de TX y RX indicarán el estado de la comunicación. El LED TX se encenderá cuando esté transmitiendo datos por el bus RS-485.

El LED RX se encenderá cuando se reciba un dato desde el bus RS-485. Tan sólo se encenderá este LED cuando el dato detectado tenga el formato serie adecuado. Si hay tráfico en el bus y la velocidad de comunicación es diferente a la configurada en el esclavo, este LED permanecerá apagado.

Como en cualquier red RS-485, se requiere una terminación de bus en ambos extremos. También, puede ser necesaria una linea de polarización para cada dispositivo. Todo esto puede ser configurado mediante los switches descritos en la sección 5.

4.2.2.- Modulo RS-232

Esta versión usa un conector macho DB9 estándar, el cual puede ser conectado al puerto serie de un PC con un cable null-modem normal. El esquema de este cable se muestra en el siguiente esquema:

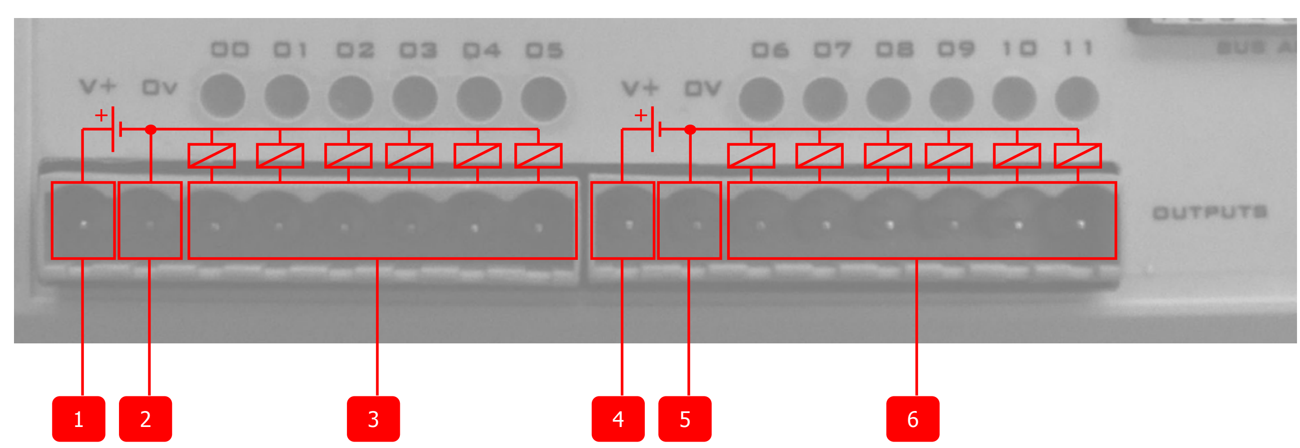


Aunque es opcional se recomienda conectar las dos carcasas DB9 entre sí para incrementar la inmunidad ante el ruido. La conexión serie con puertos serie DB25 también es posible usando adaptadores estándar DB25-DB9.

Los LEDs TX y RX LEDs indicarán el estado de la comunicación igual que en la versión con conector RS-485.

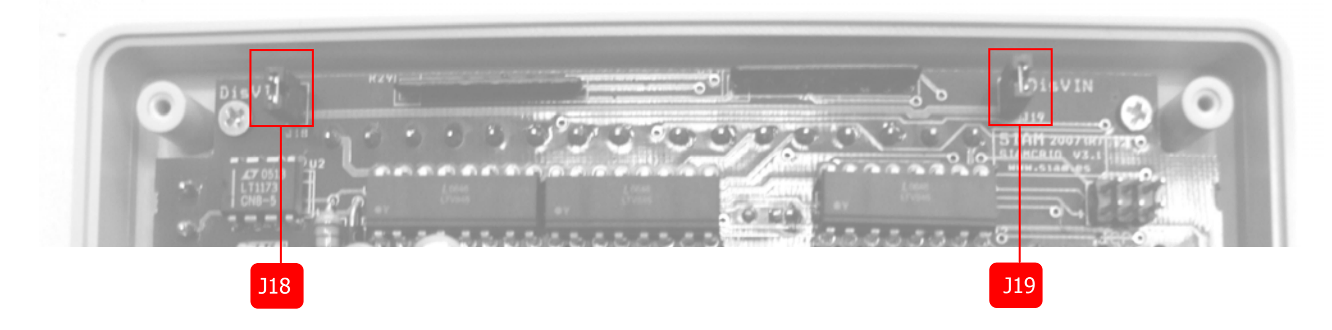
4.3.- Salidas digitales

La placa dispone de 12 salidas digitales optoacopladas con transistores Darlington PNP, con 2 fuentes de alimentación independientes, una para las salidas 0...5 y otra para las salidas 6...11. La conexión de las salidas se realiza mediante los siguientes conectores:

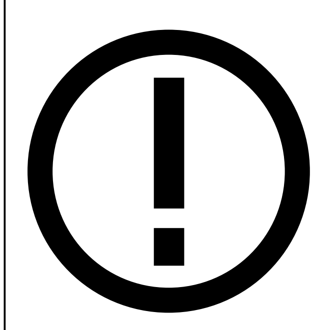


Conector	Descripción	Comentarios
1	Alimentación para las salidas digitales DO00-DO05 (9VDC…30VDC)	Conectar aquí el terminal positivo de la fuente de alimentación para el primer grupo de salidas digitales. Esta fuente de alimentación puede ser interrumpida con una parada de emergencia o con un relé de seguridad por ejemplo para desactivar este grupo de salida sin afectar a las demas.
2	Alimentación para las salidas digitales DO00-DO05 (GND)	Esta es la masa de la fuente de alimentación del primer grupo de salidas digitales. Internamente está conectada con la masa de la principal fuente de alimentación.
3	Conectores para las salidas digitales DO00-DO05	Aquí se conectan las salidas digitales DO00…DO05. Cuando una salida se active, el LED correspondiente a dicha salida se activará y la tensión será la misma que la tensión del terminal 1 (9…30VDC, normalmente 24VDC). De otra manera, el voltaje en el terminal será 0VDC y su LED estará apagado.
4	Alimentación para las salidas digitales DO06-DO11 (9VDC…30VDC)	Conectar aquí el terminal positivo de la fuente de alimentación para el segundo grupo de salidas digitales. Esta fuente de alimentación puede ser interrumpida con una parada de emergencia o con un relé de seguridad por ejemplo para desactivar este grupo de salida sin afectar a las demás.
5	Alimentación para las salidas digitales DO06-DO11 (GND)	Esta es la masa de la fuente de alimentación del segundo grupo de salidas digitales. Internamente está conectada con la masa de la principal fuente de alimentación.
6	Conectores para las salidas digitales DO06-DO11	Aquí se conectan las salidas digitales DO00…DO05. Cuando una salida se active, el LED correspondiente a dicha salida se activará y la tensión será la misma que la tensión del terminal 1 (9…30VDC, normalmente 24VDC). De otra manera, el voltaje en el terminal será 0VDC y su LED estará apagado.

El uso de fuentes de alimentación independientes para la electrónica y para las salidas, permite el uso de diferentes voltajes, por ejemplo 24VDC para la circuitería y 12VDC para las salidas. En muchos casos se usa la misma tensión de 24VDC para ambos. En estos casos la placa permite derivar la fuente principal de alimentación (terminal positivo) a los otros terminales mediante dos jumpers internos que se muestran a continuación:



Jumper	Descripción
J18	Derivación de la alimentación principal hasta la alimentación de las salidas DO00...DO05
J19	Derivación de la alimentación principal hasta la alimentación de las salidas DO06...DO11



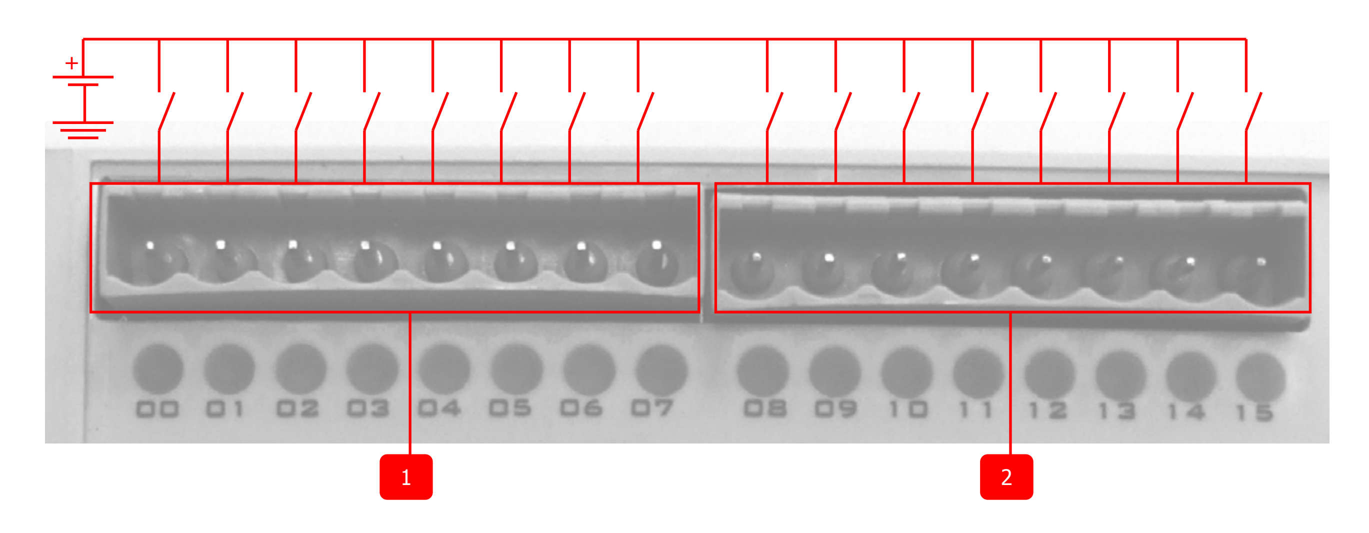
Si alguno de los jumpers de derivación está cerrado, no conectar la fuente de alimentación de los conectores de las salidas digitales (conectores 1,2,4,5 en la anterior página) a una fuente independiente. Haciendo esto cortocircuitaremos la fuente principal con la de la fuente de salidas.

Estos jumpers solamente derivan la tensión positiva de la fuente de alimentación, pues las masas ya están internamente conectadas con la fuente principal.

Las salidas digitales pueden ser activadas internamente por el programa IEC (%QW0) o externamente mediante comunicaciones modbus, escribiendo un registro de 16 bits cuyos primeros 12 bits representan las 12 salidas digitales. En principio, un bit activo (1) en el registro de salidas activa la salida correspondiente, y un bit inactivo (0) la desactiva. Si se está controlando las salidas mediante la comunicación modbus e internamente por programa simultáneamente, normalmente prevalece el control por programa debido a su mayor velocidad de ejecución.

4.4.- Entradas digitales

La placa dispone de 16 entradas digitales optoaisladas. La conexión de las entradas está hecha en los siguientes conectores:



Conector	Descripción	Comentarios
1	Conectores para las entradas digitales DI00-DI07	Aquí se conectan las primeras 8 entradas DI00…DI07. Cuando la entrada correspondiente está activa (voltaje en el terminal es 9VDC o mayor), su led se encenderá. Cuando no lo está, permanecerá apagado.
2	Conectores para las entradasDI08-DI15	Aquí se conectan las entradas DI08…DI15. Cuando la entrada correspondiente está activa (voltaje en el terminal es 9VDC o mayor), su led se encenderá. Cuando no lo está, permanecerá apagado. Estas entradas están multiplexadas con las 8 analógicas 0…7, las cuales también están conectadas al mismo conector. **Este grupo de entradas deben ser configuradas mediante jumpers hardware,** como se describe en la siguiente sección.

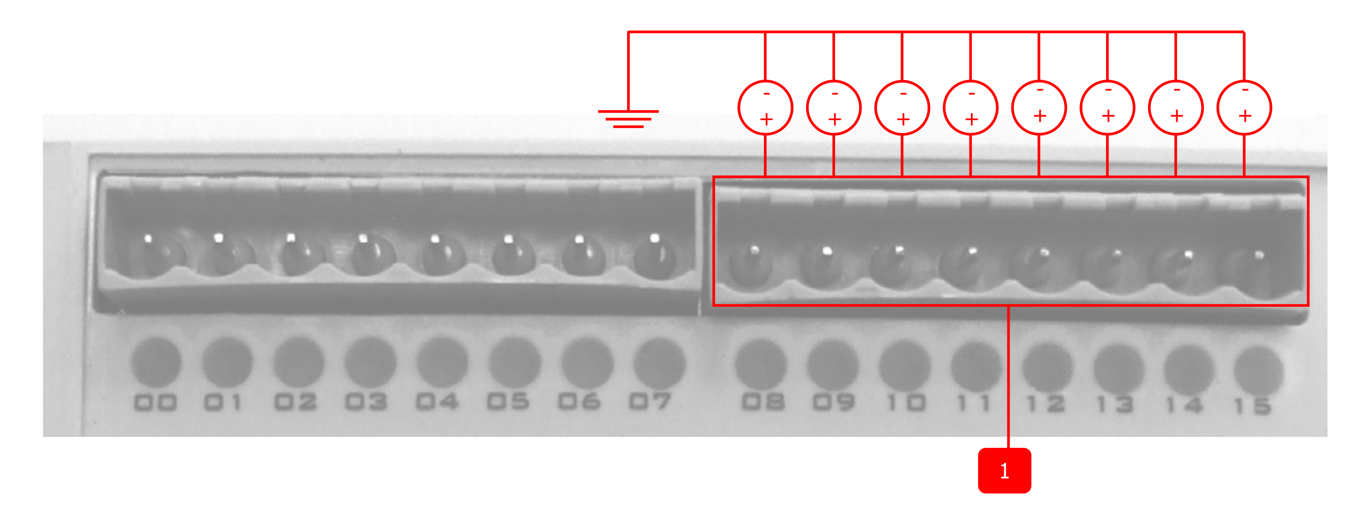
Los voltajes de las entradas digitales deben estar entre 9 y 30VDC, referenciados a la masa de la fuente de alimentación.

Se puede acceder a las entradas digitales mediante el programa de usuario IEC (%IW0) o externamente mediante comunicación modbus, leyendo un registro de 16 bits cuyos bits están mapeados a las 16 entradas digitales. Un bit activo (1) en el registro de entradas indica que la entrada correspondiente está activa, y un bit inactivo (0) indica que está inactiva.

Las entradas analógicas también están mapeadas a este registro comparando el valor analógico de las entradas. Si una entrada analógica es mayor de 2.5V o 10mA activará el bit correspondiente a dicha entrada en el registro de entradas.

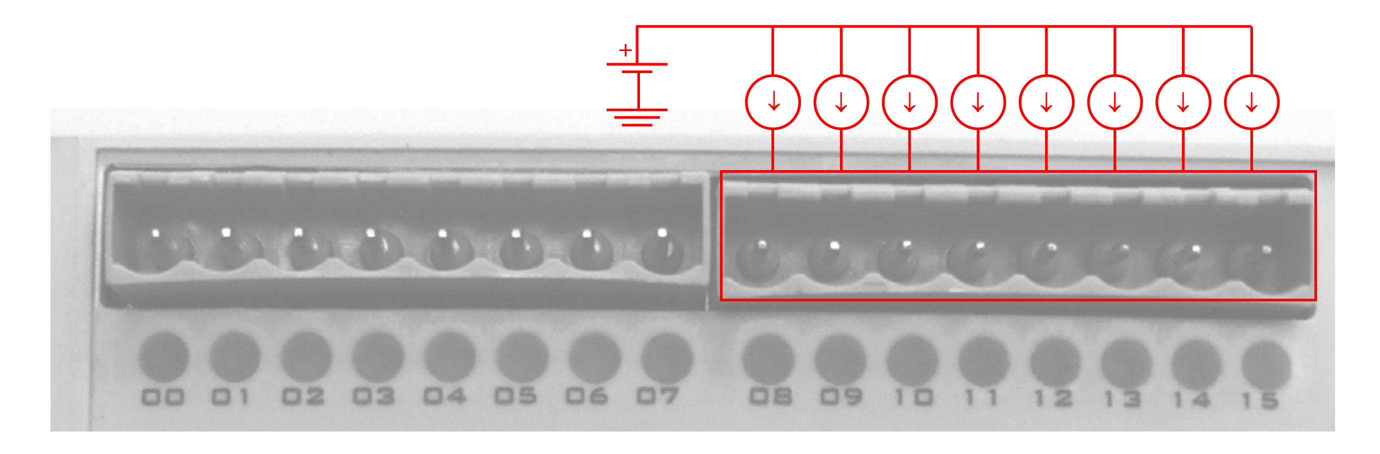
4.5.- Entradas analógicas

Las entradas analógicas comparten los mismos conectores que las entradas digitales DI08…DI15. Pueden ser entradas de tensió 0…5VDC referenciadas a la masa de la fuente de alimentación principal como se muestra aquí:



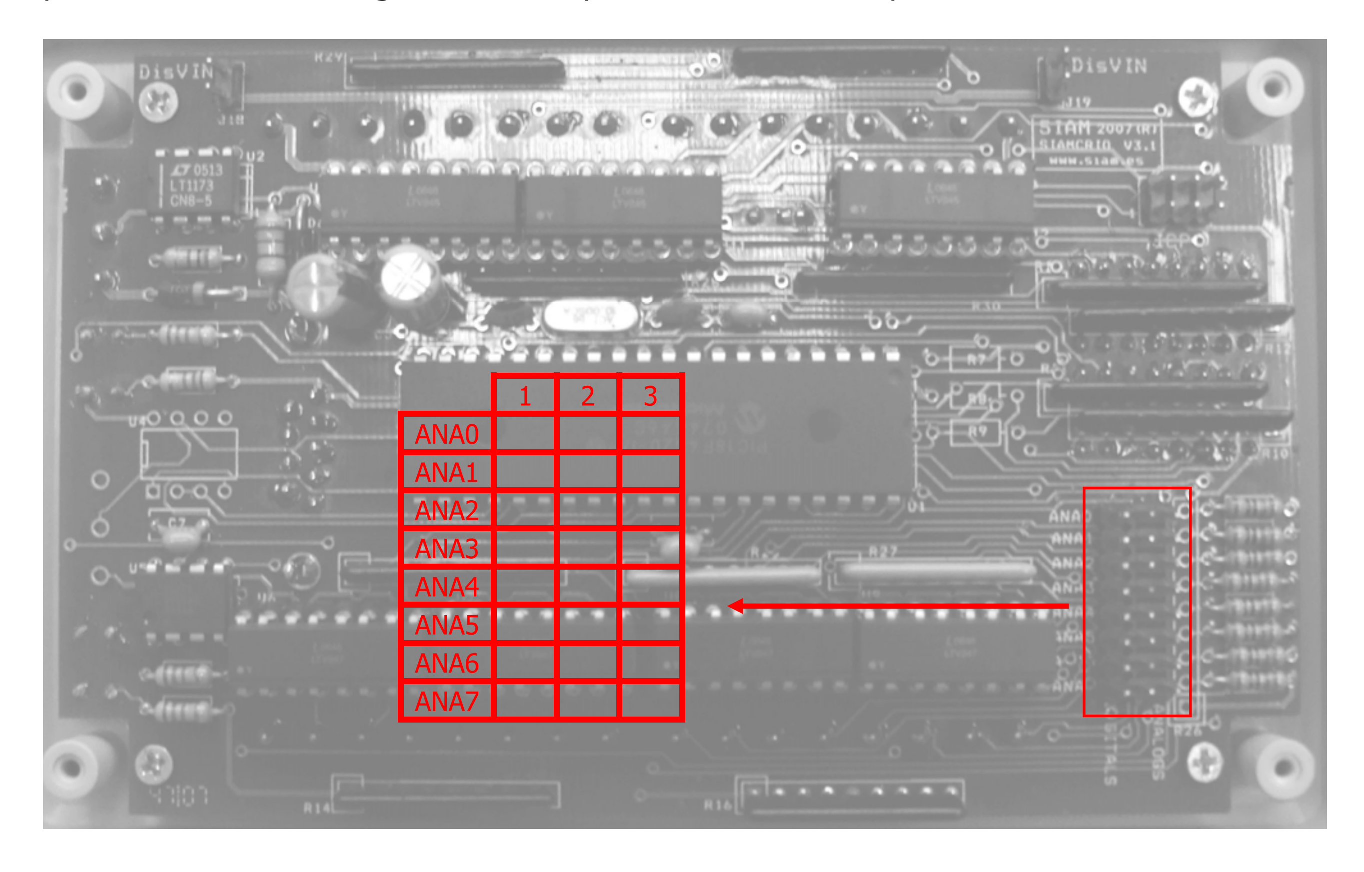
Conector	Descripción	Comentarios
1	Conector para las entradas analógicas AI00-AI07	Aqui se conectan las entradas analógicas AI00…AI07.

También pueden ser entradas de corriente 0…20mA:



En este caso, las entradas analógicas son entradas **pasivas**. Por tanto los transmisores analógicos deben tener su propia alimentación y no pueden ser alimentados mediante las entradas analógicas.

Las entradas analógicas deben ser configuradas fijando adecuadamente los jumpers dentro de la placa. Hay 8 jumpers en la placa, uno por cada una de las posibles entradas analógicas como se puede ver en este esquema:



Para configurar cualquier entrada como analógica, cerrar jumper 2-3. Para configurar cualquier entrada como digital, cerrar jumpers 1-2. Cualquier combinación de entradas digital-analógica es posible.



No conectar nunca una tensión de entrada digital (24V) a una entrada configurada como analógica. Conectar a una entrada analógica un voltaje mayor de 5V puede dañar la placa.

Al contrario, conectar un voltaje analógico (0…30V) en una entrada configurada como digital no supone un peligro para la placa.

Las entradas analógicas son convertidas con una resolución 10 bits, escaladas y mapeadas a los registros de 16 bits %IW1…%IW8, accesibles mediante el programa de usuario IEC o mediante comunicación modbus. Como la resolución de las entradas analógicas es de 10 bits, los 6 bits de mayor orden siempre están a 0. Por tanto los registros de entradas analógicas siempre contendrán un valor entre 0 y 1023. Estas entradas son también umbralizadas y mapeadas en los registros de entadas digitales (bits 7…15).

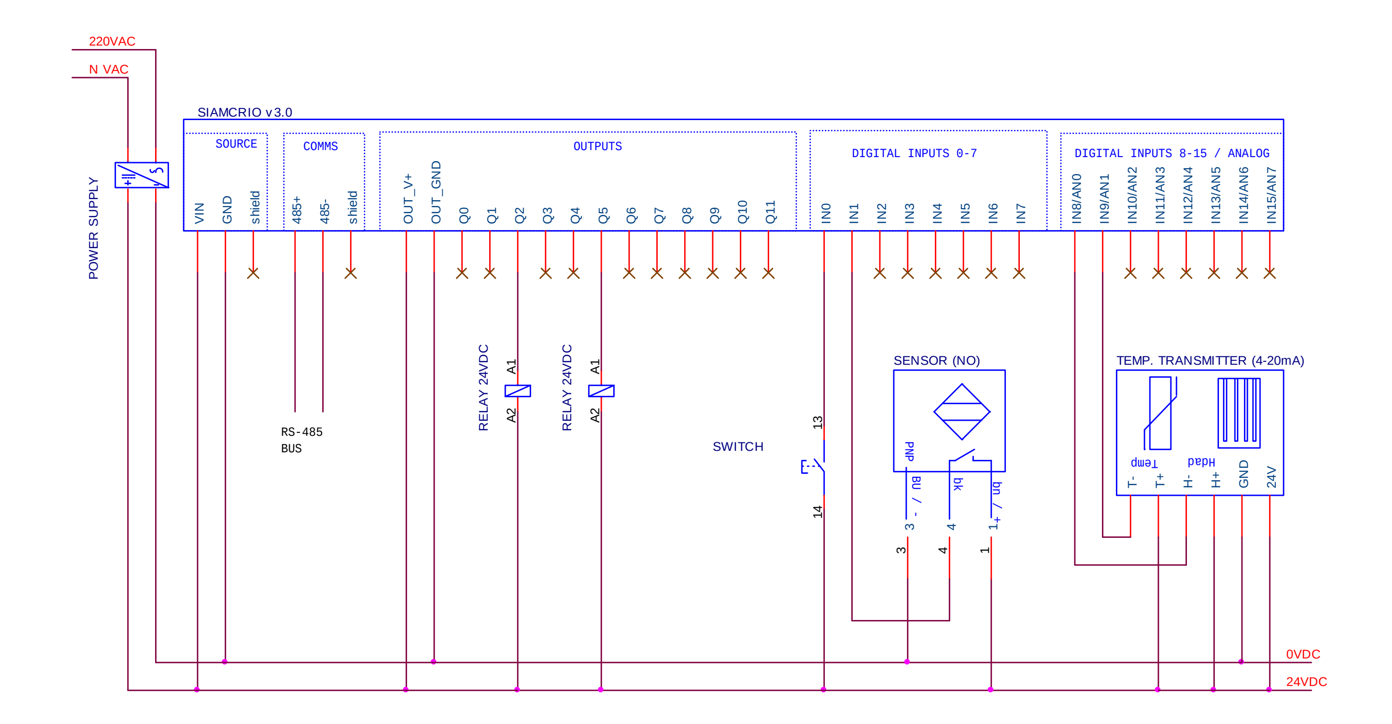
El escalado de los registros de 16 bits se realiza mediante conversión lineal, cuyo rango puede ser configurado por programa. Por defecto, el rango es de 0 a 10000 por lo que 0V/0mA serán convertidos en el valor 0, y 5V/20mA en el valor 10000. Pero este rango puede ser cambiado a cualquier valor de tipo word con signo entro los valores -32768 y +32767.

El valor de conversión de 10 bits sin escalado (0...1023) también puede ser leído mediante los registros de sistema %SW30…%SW37.

4.6.- Ejemplo de conexión

Los siguientes diagramas muestran un ejemplo de conexión de la placa:

- Las salidas %Q0.2 y %Q0.5 están comandando dos relés de 24VDC.
- La entrada %I0.0 está conectada a un interruptor (switch).
- La entrada %I0.1 esta conectada a un sensor con un contacto normalmente abierto.
- Las entradas analógicas %IW0 y %IW1 leen la humedad y la temperatura desde un transmisor analógico de 4-20mA.

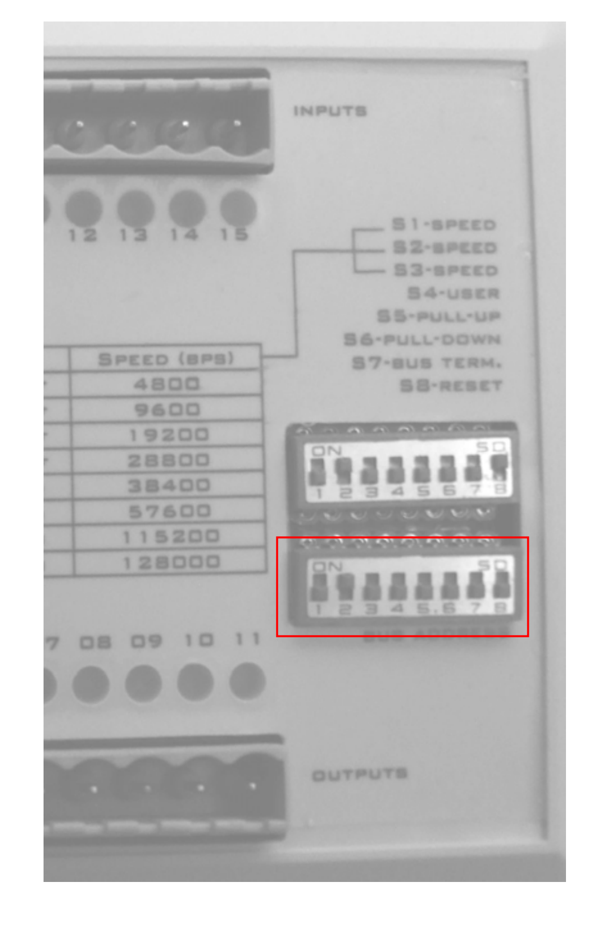




5.- CONFIGURACIÓN DEL EQUIPO

5.1.- Direccion del bus

El primer paso para conigurar la comunicación es fijar la dirección del esclavo. Esto puede ser hecho con el banco de interruptores DIP, mostrados en la figura de la derecha.



Hay 256 direcciones posibles, pero sólo las direcciones entre la 1 y la 247 son validas para el protocolo modbus. El interruptor DIP 1 contiene el bit menos significativo de la dirección (LSB), mientras que el interruptor 8 contiene el bit más significativo (MSB).

Para resetear el programa de usuario configurar la dirección con el valor 0. En caso de una incorrecta programacion o configuración, encender la placa con la dirección 0, es decir con todos los interruptores a 0, y el programa será eliminado. En este caso la dirección por defecto será fijada como la 128. El programa podrá ser corregido y descargado otra vez.

La dirección del bus debe ser configurada para ambas versiones: RS-232 y RS-485.

En el arranque, la dirección del banco de interruptores es mapeada al registro de sistema %SW72 que controla la dirección útil de la placa. Éste es un registro de lectura y escritura, por lo que el programa del usurario puede sobrescribir su valor para cambiar la dirección modbus.

La dirección del bus fijada por los interruptores no es leída continuamente, sino que solamente se lee en el arranque. Por lo que, una vez cambiada la dirección de placa, debe ser reiniciada quitándole la tensión de alimentación o accionando el interruptor S8 en el banco de interruptores superior como se explica abajo.

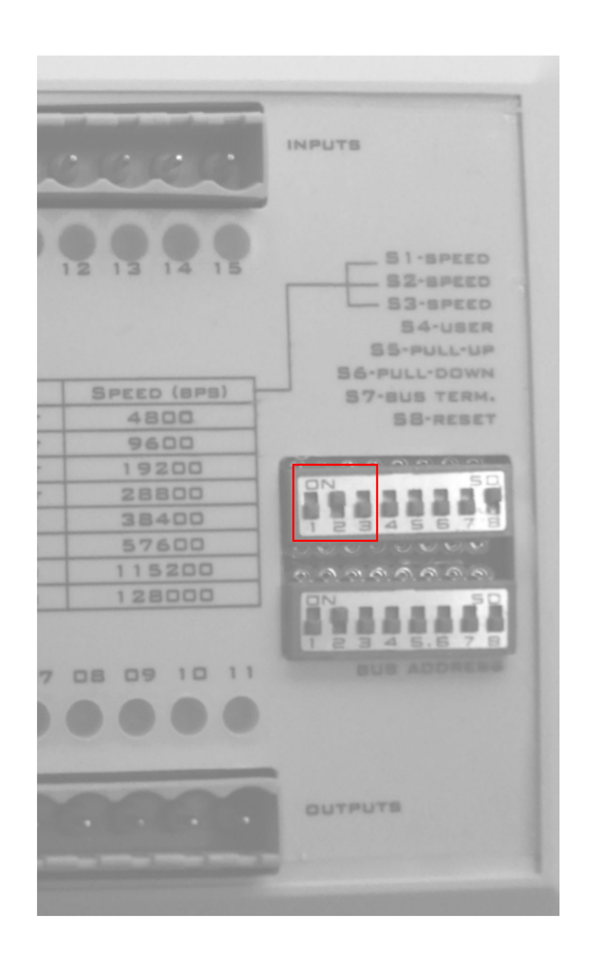
5.2.- Velocidad de transferencia

La velocidad de comunicación es seleccionada mediante los interruptores 1-2-3 del banco DIP, mostrado en la imagen de la derecha.

Las siguientes velocidades pueden ser elegidas:

S1	S2	S3	Velocidad
OFF	OFF	OFF	4800 bps
OFF	OFF	ON	9600 bps
OFF	ON	OFF	19200 bps
OFF	ON	ON	28800 bps
ON	OFF	OFF	38400 bps
ON	OFF	ON	57600 bps
ON	ON	OFF	115200 bps
ON	ON	ON	128000 bps

Como en el anterior caso, la selección de la velocidad de transmisión no es leída continuamente, sino que sólo en el arranque. Así pues, se requiere reiniciar la placa después de cambiar la dirección.



5.3.- Interruptor de usuario

El interruptor 4 del banco superior no tiene uso predefinido. Es mapeado en el arranque al bit del registro de sistema %S6.11 (todos los interruptores del banco superior son mapeados al registro de sistema %SW6). El programa de usuario IEC puede comprobar este bit para cualquier propósito, o ignorarlo.

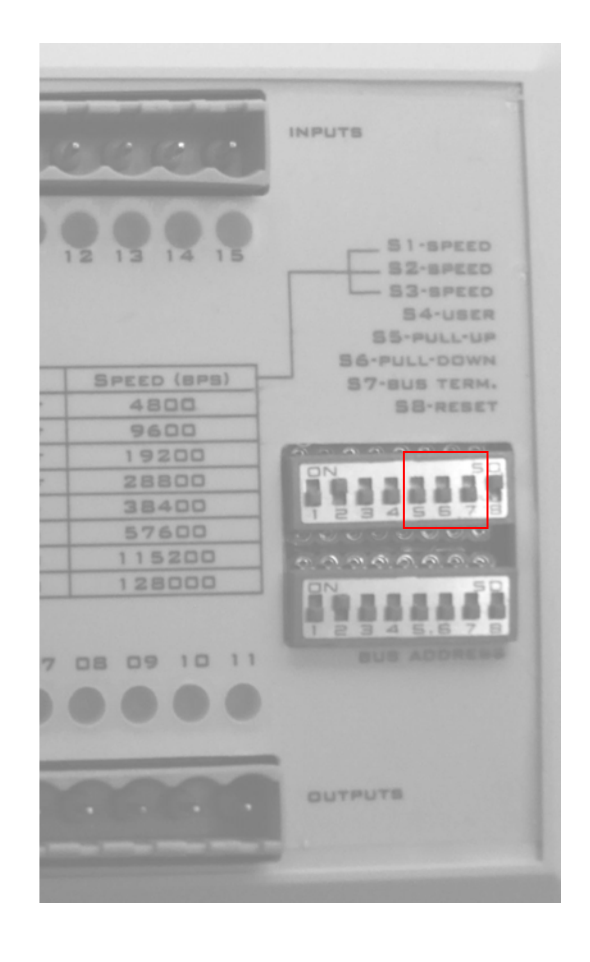
Como el resto de interruptores, tan sólo será leido en el arranque, con lo que es necesario reiniciar la placa después de un cambio.



5.4.- Resistencias de comunicación

Los interruptores 5-6-7 del banco DIP superior, de la imagen de la derecha, controla las resistencias de comunicación para el bus RS-485:

Jumper	Description	Comments
S5	Resistencia **Pull-up** (390 Ω)	Estabiliza el bus. **Debe estar activa sólo en un nodo de la red RS-485.**
S6	Resistencia **Pull-down** (390 Ω)	Estabiliza el bus. **Debe estar activa sólo en un nodo de la red RS-485.**
S7	**Resistencia de terminación de bus** (220 Ω)	Previene reflexiones de bus. **Debe estar activa sólo en los extremos del bus**, e inactivo en todos los demas nodos intermedios.



Estos interruptores son interruptores hardware, así que pueden ser activados o desactivados en cualquier momento (no sólo en el arranque)

Estos interruptores no se usan en la version RS-232 de la placa.

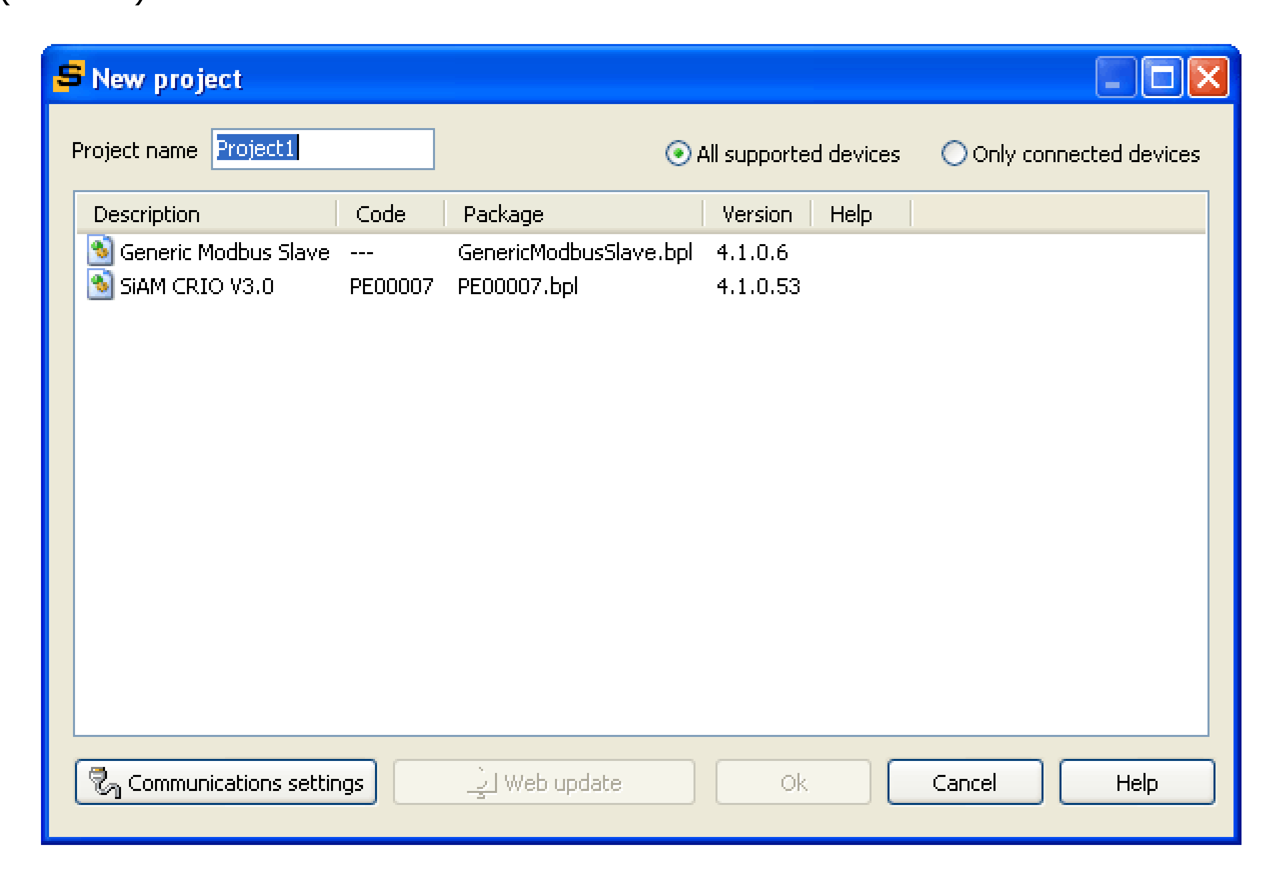
5.5.- Interruptor de reset

El interruptor 8 del banco DIP superior se usa para hacer un reset hardware de la placa sin necesidad de tener y que desconectar y conectar la fuente de alimentación. Si este interruptor permance OFF el microcontrolador será constantemente reseteado.

Esto puede ser muy útil y rápido para actualizar la configuración de la placa –dirección modbus, velocidad de transmision- después de cualquier cambio de configuración en los interruptores.

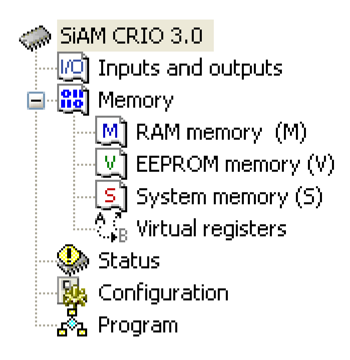
6.- SOFTWARE

El software para la configuración y programación de controlador SiAMCRIO se llama CRIOSoft, el cual puede ser descargado gratuitamente de la página web de SIAM. Una vez ejecutado CRIOSoft, abrir un nuevo proyecto y seleccionar el dispositivo **SiAM CRIO v3.0** (PE00007).



Si la placa está conectada a la red modbus y el puerto serie está correctamente configurado, seleccionar **Only connected devices** y pulsar **Search** para escanear el bus. Si la placa es encontrada, hacer doble click sobre ella para abrir un nuevo proyecto.

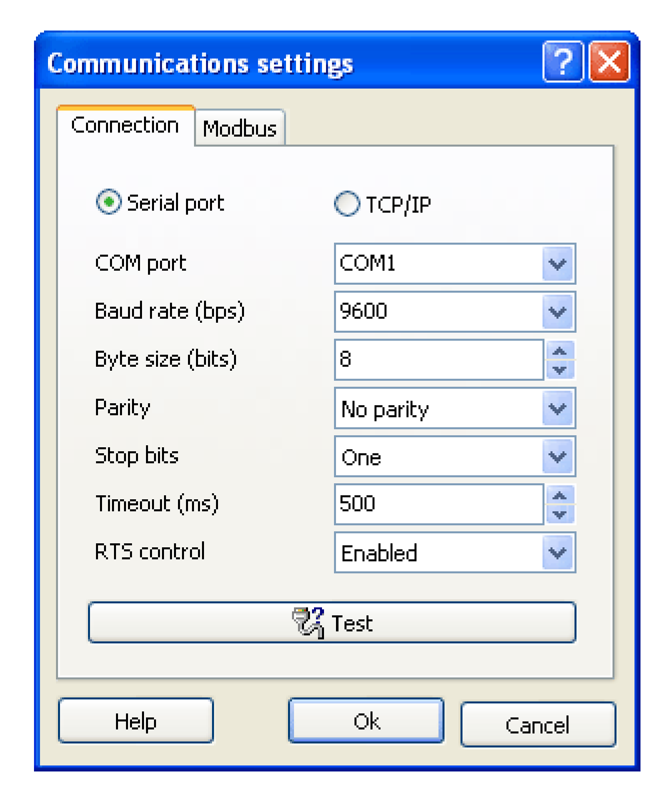


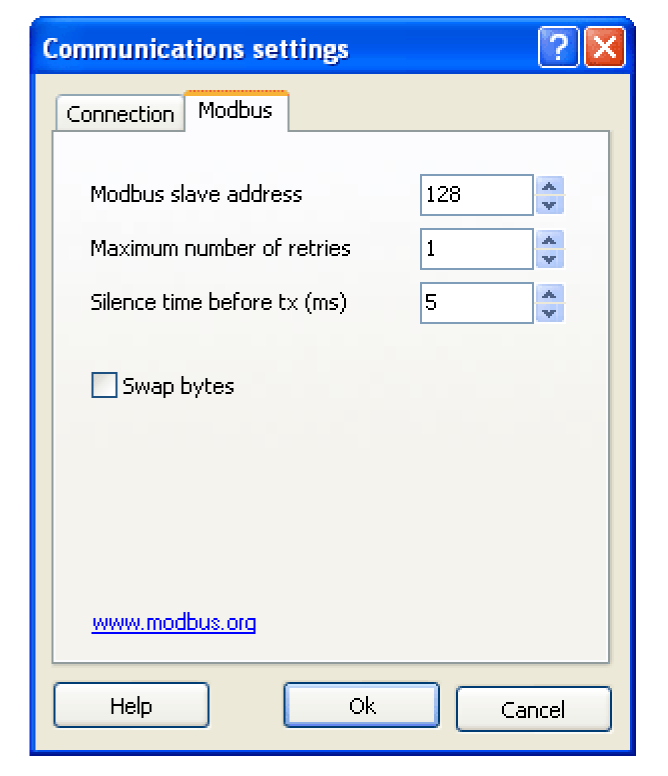


Las principales opciones del dispositivo aparecerán en el panel de la izquierda de la pantalla de CRIOSoft, ademas de otros botones sobre la barra de arriba y de menus con con más opciones.

Lo primero a hacer es configurar la comunicación, para ello seleccionaremos **Communication settings** en el menú **Tools**. Los parámetros seleccionados aquí tienen que coincidir con los fijados mediante los interruptores de la placa, explicados en la sección anterior (dirección y velocidad de transmisión).

El formato de datos para comunicación serie debe ser de **8 bits de datos**, **1 bit de stop** y **sin paridad**.



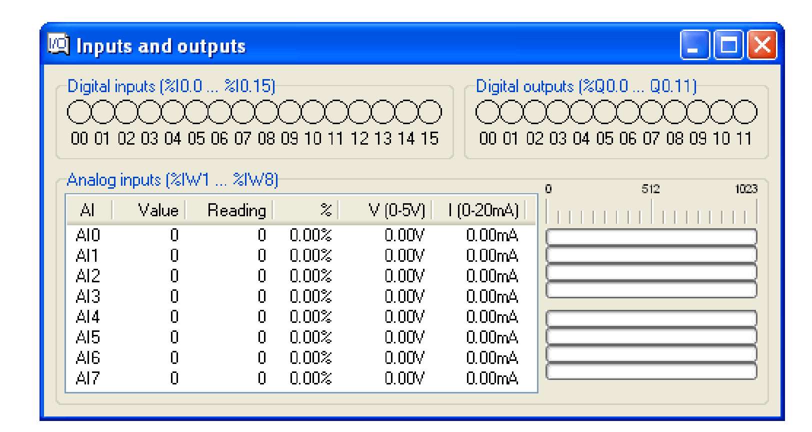


6.1.- Modo online

La monitorización de señales, entradas, salidas registros, etc… solamente es posible en modo **online**. Para conectarse en modo online, seleccionar **Monitor** en el menu **Device**, o hacer click en el botón .

6.2.- Entradas y salidas

La primera opcion despliega esta ventana, que muestra el estado de las entradas y salidas en modo **online**.



Para cualquier entrada analogica se puede ver:

- El **Valor** escalado
- **Lectura** sin escalar (0-1023)
- Porcentaje del valor (0-100%)
- Valor del voltaje y de la corriente.

El estado de las salidas pueden ser cambiadas haciendo clic sobre ellas con el botón derecho del ratón.

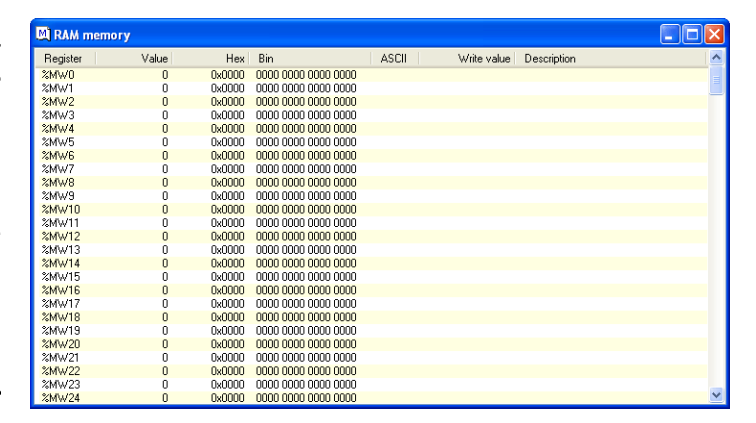
El estado de las señales digitales es el correspondiente a los registros internos de la placa, y por tanto puede estar afectado por las mascaras de forzado, lo que significa que tanto las entradas como las salidas digitales pueden ser forzadas. Cualquier entrada o salida forzada mostrara un **F** sobre ella.

Para forzar una entrada o una salida, estando siempre en modo online, pulsaremos el botón derecho del ratón sobre el indicador correspondiente. Nos aparecerán opciones para forzar a ON y a OFF, así como para eliminar este forzado o todos ellos. Por último nos aparecerá una opción que podemos marcar y desmarcar para habilitar o deshabilitar las mascaras de forzado.

Los forzados son volátiles, esto es, si se apaga la placa se pierden las mascaras de forzado. Los forzados persistentes de entradas o salidas se deben hacer el mediante programa IEC.

6.3.- Memoria RAM (M)

La siguiente opción nos muestra los registros volátiles de la memoria M de la placa en formatos decimal, hexadecimal, binario y ASCII.



Register	Value	Hex	Bin	ASCII	Write value	Description
%MW0	0	0x0000	0000 0000 0000 0000			
%MW1	0	0x0000	0000 0000 0000 0000			
%MW2	0	0x0000	0000 0000 0000 0000			
%MW3	0	0x0000	0000 0000 0000 0000			
%MW4	0	0x0000	0000 0000 0000 0000			
%MW5	0	0x0000	0000 0000 0000 0000			
%MW6	0	0x0000	0000 0000 0000 0000			
%MW7	0	0x0000	0000 0000 0000 0000			
%MW8	0	0x0000	0000 0000 0000 0000			
%MW9	0	0x0000	0000 0000 0000 0000			
%MW10	0	0x0000	0000 0000 0000 0000			
%MW11	0	0x0000	0000 0000 0000 0000			
%MW12	0	0x0000	0000 0000 0000 0000			
%MW13	0	0x0000	0000 0000 0000 0000			
%MW14	0	0x0000	0000 0000 0000 0000			
%MW15	0	0x0000	0000 0000 0000 0000			
%MW16	0	0x0000	0000 0000 0000 0000			
%MW17	0	0x0000	0000 0000 0000 0000			
%MW18	0	0x0000	0000 0000 0000 0000			
%MW19	0	0x0000	0000 0000 0000 0000			
%MW20	0	0x0000	0000 0000 0000 0000			
%MW21	0	0x0000	0000 0000 0000 0000			
%MW22	0	0x0000	0000 0000 0000 0000			
%MW23	0	0x0000	0000 0000 0000 0000			
%MW24	0	0x0000	0000 0000 0000 0000			

Pulsando el botón derecho sobre cualquierregistro nos aparecerán opciones para cambiar el valor del registro, borrarlo (ponerlo a 0) y para asignarle una descripción textual. Los cambios de valor de registro se almacenan y no se hacen efectivos hasta que se entra en modo online.

6.4.- Memoria EEPROM (V)

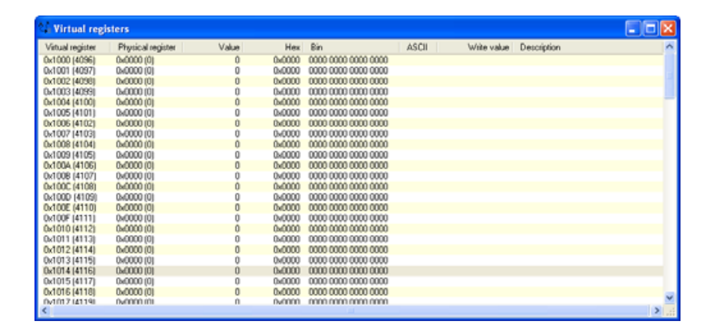
Esta opción nos muestra una ventana como la anterior donde podremos ver y modificar el valor de los registros no volátiles de la memoria V de la placa.

6.5.-Memoria de sistema

Esta opción nos muestra una ventana como la anterior donde podremos ver (pero no modificar) el estado de los registros de la memoria S de la placa.

6.6.- Registros virtuales

Esta opción nos muestra una ventana similar a las anteriores donde podremos ver y modificar el estado de los registros virtuales de direccionamiento indirecto de la placa.



Virtual register	Physical register	Value	Hex	Bin	ASCII	Write value	Description
0x1000 (4096)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x1001 (4097)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x1002 (4098)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x1003 (4099)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x1004 (4100)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x1005 (4101)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x1006 (4102)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x1007 (4103)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x1008 (4104)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x1009 (4105)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x100A (4106)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x100B (4107)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x100C (4108)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x100D (4109)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x100E (4110)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x100F (4111)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x1010 (4112)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x1011 (4113)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x1012 (4114)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x1013 (4115)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x1014 (4116)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x1015 (4117)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			
0x1016 (4118)	0x0000 (0)	0	0x0000	0000 0000 0000 0000			

Para cada registro modbus virtual, podemos ver el registro modbus real al que hace referencia, así como el valor del registro real en formatos decimal, hexadecimal, binario y ASCII. También podemos asignar descripciones textuales a los registros virtuales.

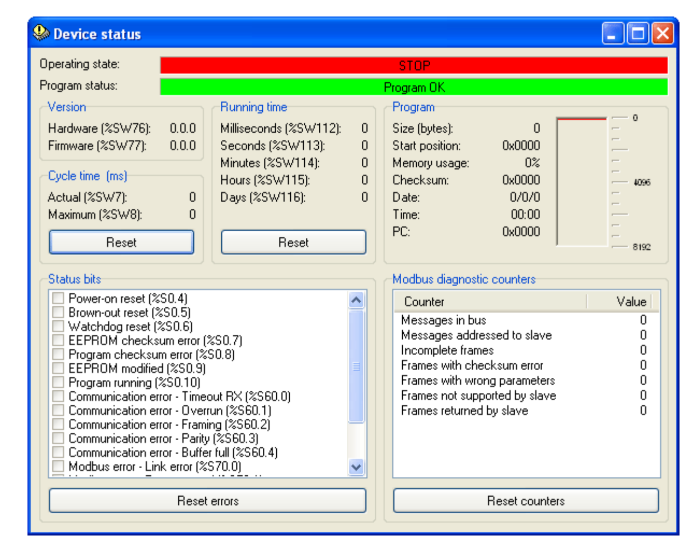
Pulsando el botón derecho sobre cada registro podemos, como en los casos anteriores, cambiar el valor del registro real. Ademas podemos cambiar la asignación a registros físicos, pulsando sobre la opción **Change physical register**.

Esta tabla puede ser muy útil para monitorizar de forma eficiente variables de programa que se encuentran en posiciones dispersas o en distintas áreas de memoria, ya que toda la zona de registros virtuales se refresca muy rápidamente, a diferencia de las multiples lecturas necesarias para poder leer áereas de memorias dispersas. Ver sección 6.8.2 para más información sobre registros virtuales.

La asociación de registros físicos a registros virtuales es almacenada en la memoria flash, con lo que pemanecerán aunque se quite la alimentación.

6.7.- Estado

La ventana de estado muestra información importante sobre el estado actual de la placa:



- Estado operativo (RUN o STOP)
- Estado del programa (OK o con error)
- Hardware y version del firmware de la placa.
- Tiempo de ciclo actual y maximo.
- Estado bits importantes.
- Contadores del tiempo de funcionamiento.
- Estadisticas de programa de usuario, y grafico de ocupación de memoria.

Disponemos en esta ventana de botones para resetear el tiempo de ciclo máximo, eltiempo de funcionamiento y los errores importantes.

Si hubiera algún error de programa activo, y siempre que el programa offline (el programa en el CRIOSoft) coincida con el programa cargado en la placa, podemos hacer clic sobre el panel indicador del error para abrir a la ventana de programa y saltar a la instrucción en la que se ha producido el error.

6.8.- Configuración

Los parametros más importantes para la configuración de la placa pueden ser modificados desde esta pantalla. En parte inferior de la ventana de configuración disponemos de los botones **Read** (leer) and **Write** (escribir) la configuración actual y así como para aceptar (**Ok**) o cancelar (**Cancel**) los cambios realizados. Estos parámetros solamente pueden cambiarse en modo offline.

6.8.1.- Comunicación y programación

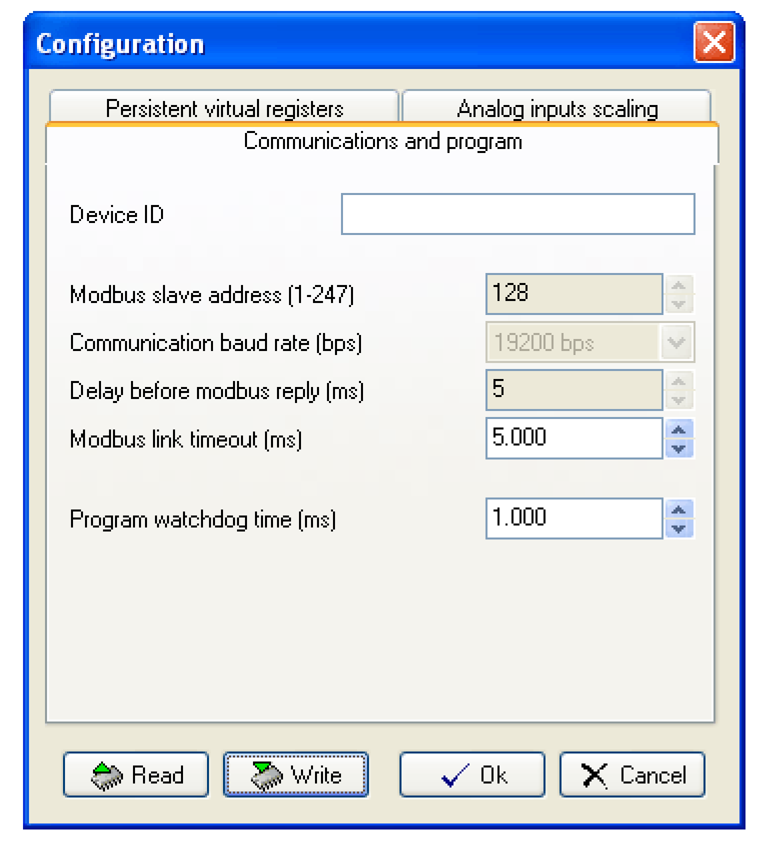
En la primera pestaña, podemos ajustar los parametros relacionados con las comunicaciones y al programa de usuario:

- Identificador del dispositivo (**Device ID**). Es un nombre de hasta 26 caracteres que identifica al esclavo modbus al hacer una busqueda mediante **Only connected devices** en las opciones de **New Project.**

- Dirección de esclavo Modbus (**Modbus slave address**). Este valor es fijado mediante el banco de interruptores visto en la sección 5.1

- Velocidad de transmisión de datos (**Communication baud rate**). Este valor es fijado mediante el banco de interruptores visto en la sección 5.2

- Retardo antes de la respuesta (**Delay before modbus reply**). Este tiempo (en milisegundos) indica tiempo de espera antes de responder a las peticiones del master modbus. Si el master necesita mucho tiempo para hacer el cambio de sentido de la transimisión –por ejemplo, en el caso de un PC con que utiliza un conversor RS232/485 con control software de flujo, especialmente a bajas velocidades (4800bps o menores)- puede que sea necesario aumentar el tiempo.



- **Modbus link timeout**. Si este tiempo es superado sin recibir ninguna petición modbus, entonces la placa considerará que el enlace de comunicación se ha perdido. El estado del enlace de comunicación será mapeado al bit de sistema%S70.0. El

programa de usuario puede usar este bit para reaccionar en caso de pérdida de comunicación

- Tiempo de vigilancia de programa (**Program watchdog time**). Este es el tiempo máximo permitido para la ejecución del programa de usuario. Si la ejecución se prolonga más de lo que especifica este parámetro, entonces se producirá un error fatal que detendrá la ejecución y apagará todas las salidas digitales.

6.8.2.- Registros virtuales persistentes

En el segundo panel configuramos el **direccionamiento indirecto** modbus. Esto es de gran utilidad para optimizar el tiempo de ciclo de las comunicaciones serie.

El direccionamiento indirecto permite leer o escribir un registro modbus real o físico a través de un registro virtual o indirecto. Esto permite leer de forma contigua –con una sola trama modbus- registros dispersos por el mapa de memoria que de otra manera sólo podrían leerse mediante varias tramas, de forma mucho más lenta.

Vamos con un ejemplo con el siguiente mapeado:

- Registro físico 0x0000 -> Virtual 0x1000
- Registro físico 0x0100 -> Virtual 0x1001
- Registro físico 0x0200 -> Virtual 0x1002

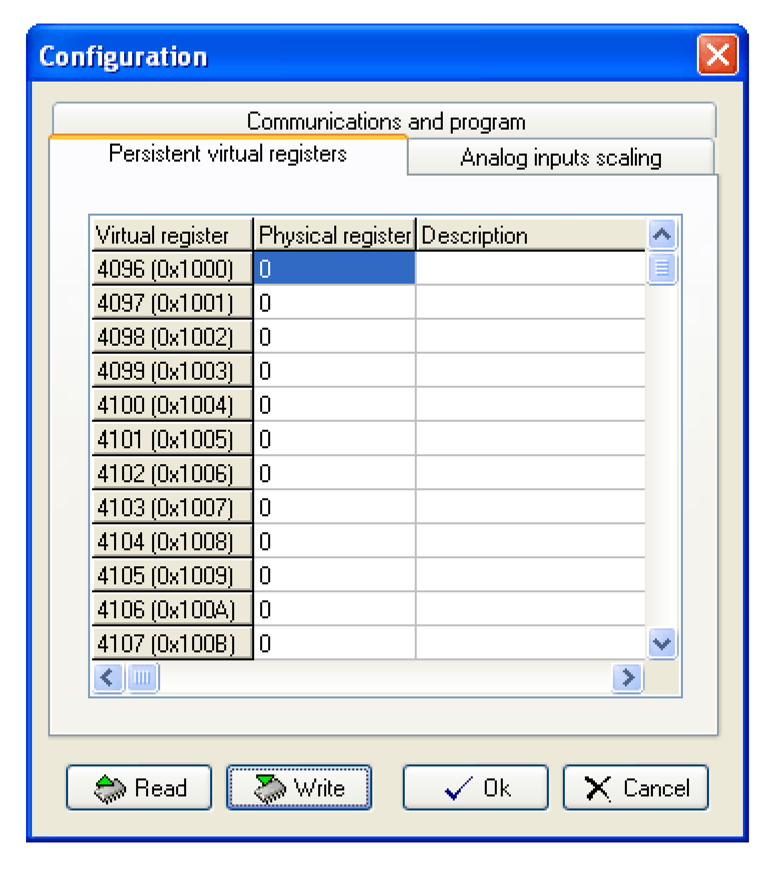
Podemos leer los registros 0x1000, 0x1001 y 0x1002 con una sola trama leyendo 3 registros contiguos, por ejemplo mediante la trama modbus READ HOLDING REGISTERS (código 0x03) y trabajar con ellos como si de los registros 0x0000, 0x0100 y 0x0200 se tratara. De otra forma, necesitaríamos 3 tramas distintas -con su petición y respuesta- y el triple de tiempo para leer estos 3 registros de forma independiente.
Cuando trabajemos con registros virtuales, realmente estaremos trabajando con su registro fisico asociado.

Este mapeado o asiciación entre registros reales y virtuales son persistentes, con lo que no se pierden al quitar la alimentación .

En la ventana de configuración de CRIOSoft podemos configurar de forma muy sencilla los 64 registros de direccionamiento indirecto de que se dispone, con direcciones entre 0x1000 y 0x103F. Podemos también asignar un comentario descriptivo a cada registro virtual.

Esta zona de direccionamiento indirecto se almacena en memoria no volátil, y por tanto sólo necesita ser configurada una vez.

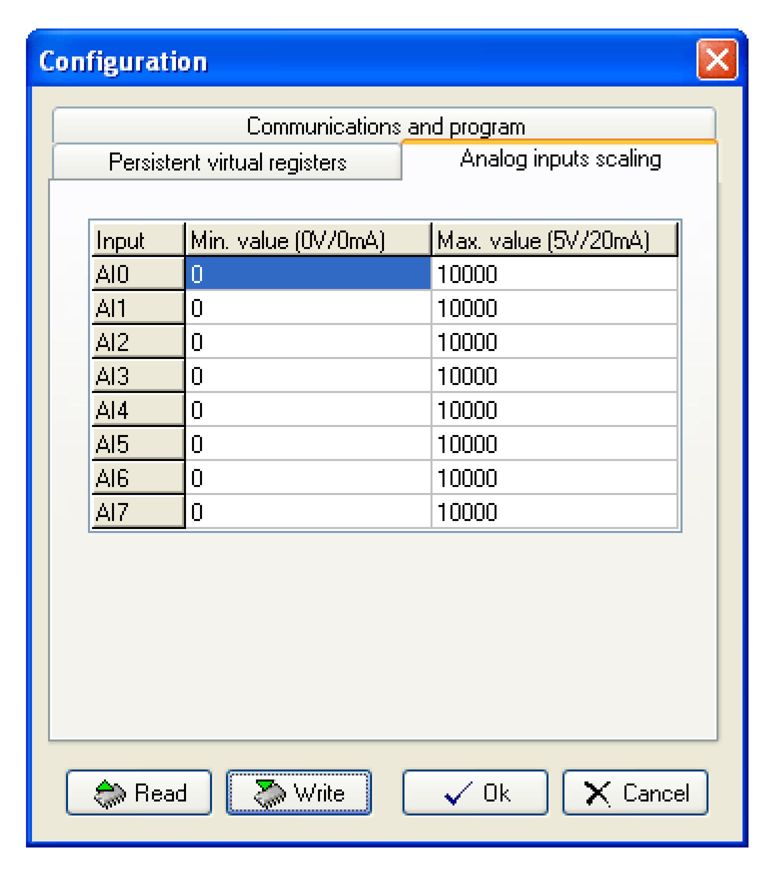
Se dispone otros 64 registros de zona de direccionamiento indirecto en el rango 0x1100-0x113F cuya configuración es volátil, y permiten hacer un direccionamiento indirecto variable en el tiempo con mucha facilidad.



6.8.3.- Escalado de las entradas analógicas

En la ultima pestaña nos encontramos con la configuración del escalado para las entradas analógicas

Para cada entrada debemos introducir el valor mínimo y máximo entre los que se realizará el escalado. Estos valores tienen que estar en el rango -32768… +32767, y el valor máximo debe ser mayor o igual que el mínimo.



Este escalado es muy útil para poder trabajar en el programa de usuario con unidades de ingeniería, en lugar que con unidades ficticias 0-1023.

Por ejemplo, si disponemos de una sonda de temperatura con salida 0-20mA y con rango de funcionamiento -40…80 ºC, podemos ajustar los valores mínimo y máximo a -4000 y 8000 respectivamente. De esta manera en el registro de la primera entrada analógica tendremos la temperatura real multiplicada por 100.

Hay que asegurarse de la correcta configuración de los jumpers internos -como se ha descrito en la sección 4.5- porque la conexión de voltaje alto (24VDC por ejemplo) a cualquier entrada configurada como analógica puede dañar la placa.



6.9.- Programa

La ventana de programa nos permite editar el programa de usuario en lenguaje de lista de instrucciones, y hacer la monitorización del mismo en modo online.

```
Program
VAR_GLOBAL

        /* Entradas */
        I_CEN_OPEN AT %I0.0: BOOL;                    /* Cenitales Abiertos Inv. 1*/
        I_CEN_CLOSED AT %I0.1: BOOL;                  /* Cenitales Cerrados Inv. 1 */
        I_ANEM_OK AT %I0.2: BOOL;                     /* Anemometro OK */
        I_SEL_CENTRAL_CENIT AT %I0.3: BOOL;           /* Selector Control central Cenitale
        I_SEL_CENTRAL_FRIO_CALOR AT %I0.15: BOOL;     /* Selector Control central Frio/Cal
        I_SEL_CENTRAL_FOTOP AT %I0.5: BOOL;           /* Selector Control central Fotoperi
        I_OPEN_CENIT_LOCAL AT %I0.6: BOOL;            /* Control Local Abrir cenitales Inv
        I_CLOSE_CENIT_LOCAL AT %I0.7: BOOL;           /* Control Local Cerrar cenitales In
        I_FRIO_LOCAL AT %I0.8: BOOL;                  /* Control Local Frio Inv. 1 */
        I_CALOR_LOCAL AT %I0.9: BOOL;                 /* Control Local Calor Inv. 1 */
        I_FOTOP1_LOCAL AT %I0.10: BOOL;               /* Control Local Fotoperiodo 1 Inv. .
        I_FOTOP2_LOCAL AT %I0.11: BOOL;               /* Control Local Fotoperiodo 2 Inv. .
        I_TERMICOS_OK AT %I0.12: BOOL;                /* Termicos OK Inv. 1 */
        I_TENSION_24V_OK AT %I0.13: BOOL;             /* Tension 24V2 Salidas OK */

        /* Salidas */
        Q_OPEN_CENIT AT %Q0.0: BOOL;                  /* Abrir Cenital Inv. 1 */
        Q_CLOSE_CENIT AT %Q0.1: BOOL;                 /* Cerrar Cenital Inv. 1 */
        Q_EXTRACTOR AT %Q0.2: BOOL;                   /* Extractor Inv. 1 */
        Q_COOLER AT %Q0.3: BOOL;                      /* Bomba cooler Inv. 1 */
        Q_CALEFACCION AT %Q0.4: BOOL;                 /* Bomba calefaccion Inv. 1 */
        Q_FOTOP1 AT %Q0.5: BOOL;                      /* Fotoperiodo 1 Inv. 1 */
        Q_FOTOP2 AT %Q0.6: BOOL;                      /* Fotoperiodo 2 Inv. 1 */
        Q_FALLO_COM AT %Q0.7: BOOL;                   /* Fallo Comunicación PLC Inv. 1 */

        /* Zona de lectura de PLC */
        CENIT_ABRIENDO AT %M0.0: BOOL;                /* Cenitales Abriendo Inv.1 */
        CENIT_ABIERTOS AT %M0.1: BOOL;                /* Cenitales Abiertos Inv.1 */
        CENIT_CERRANDO AT %M0.2: BOOL;                /* Cenitales Cerrando Inv.1 */

Line 1, Col 1                Offline program editing
```

En la parte inferior de esta ventana podemos observar el número de línea en el que nos encontramos, así como el uso de memoria del programa compilado y su tamaño (en instrucciones y bytes), si la compilación se ha llevado a cabo con éxito, el checksum y el actual estado del programa (offline o online).

Tras teclear el program IEC, el programa debe ser compilado seleccionando la opción de **Compile** del menú **Device**, o pulsando el botón . Si el programa tiene errores, estos aparacerán en la parte inferior del CRIOSoft. Si no los tiene, podremos transferir el programa seleccionando la opción **Download** del menú **Device**, haciendo clic sobre .



El programa puede ser monitorizado entrando en modo online, pero para que esto sea posible el programa en el CRIOSoft y en la placa SiAMCRIO deben ser idénticos. Si no lo son, un mensaje nos indicará que la monitorización online no es posible al entrar en modo online. En este caso, la ventana de programa aparecerá deshabilitada y el programa no podrá ser editado hasta que se vuelva a modo offline.

Si los programas en PC (CRIOSoft) y PLC (SiAMCRIO) son iguales, entonces la ventana de programa cambiará de aspecto al entrar en modo **online**, tal y como ilustra esta imagen:

Program

Line	Code	Operand	RLO	ACC
244	/*Salta al final si esta en modo aprendizaje			
245	LD APREN_COMENZADO	1	0	0
246	JMPC FIN	546	0	0
247				
248	/*Si esta abierto poner el tiempo total*/			
249	LD I_CEN_OPEN	1	0	0
250	JMPC CEN_OPEN	534	0	0
251				
252	/*Si esta cerrado poner el tiempo a 0*/			
253	LD I_CEN_CLOSED	1	0	0
254	JMPC CEN_CLOSE	526	0	0
255				
256	/*Incrementa Timer si esta abriendo*/			
257	LD FLAG_1S	1	0	0
258	AND CENIT_ABRIENDO	1	0	0
259	AND(	1	1	0
260	LD T_ACUM_CENIT	0	1	0
261	LT T_TOTAL_CENIT	254	1	0
262	)	254	0	0
263	JMPC INCREM_TIMER	493	0	0
264				
265	/*Decrementa Timer si esta cerrando*/			
266	LD FLAG_1S	1	0	0
267	AND CENIT_CERRANDO	1	0	0
268	AND(	1	1	0
269	LD T_ACUM_CENIT	0	1	0
270	GT 0	0	0	0
271	)	0	0	0
272		501	0	0

Line 244, Col 1 | Size: 171 instructions, 548 bytes | Checksum: 0x4B1D | Online program monitoring

En modo online podemos observar para cada línea de programa que contenga instrucciones ejecutables los siguientes datos:

- Valor del operando.
- Valor del acumulador lógico (RLO).
- Valor del acumulador numerico (ACC).

Estos valores mostrados son leidos justo en el momento despues de la ejecución de la actual instrucción.

6.10.- Otras opciones

En el menú de **Program** y en la barra superior de botones se dispone de otras opciones muy importantes para la programación de la placa SiAM CRIO, que son las siguientes:

- **Compile**: Compila el programa de usuario. Si hay errores o avisos, estos aparecerán en la parte inferior de la pantalla del CRIOSoft, y haciendo doble clic sobre ellos se resaltará la línea de código donde se encuentra el problema. Es necesario compilar el programa antes de transferirlo a la placa.

- **Download**: Selecciona esta opción para volcar el programa compilado a la placa SiAM CRIO. Para poder transferir, debemos estar offline y el programa debe haber sido compilado correctamente. Si el programa a transferir tiene un tamaño reducido y cabe en la memoria disponible junto con el programa que se ejecuta actualmente, se conmutará automáticamente al nuevo programa después de ser transferido. Si, por el contrario, el programa a transferir tiene un tamaño considerable, será necesario pasar a STOP durante la transferencia. Al concluir, se preguntará al usuario si se desea pasar a modo RUN para ejecutar el nuevo programa transferido.

- **Compare**: Esta opción muestra si el programa en el PLC (placa) y en el PC (CRIOSoft IDE) son iguales.

- **Memory erase**: Permite hacer un borrado de la memoria de programa de la placa y dejarla vacía. Es necesario estar en modo STOP.

- **RUN**: Arranca la ejecución del programa. En el área inferior del CRIOSoft se indicará si se ha conseguido arrancar con éxito, o por el contrario, si se ha producido un error.

- **STOP**: Detiene la ejecución del programa. Puede ser necesario pasar a modo STOP para transferir un nuevo programa o para hacer un borrado de memoria. Al parar el programa se desactivarán todas las salidas.

7.- Programación IEC

7.1.- El lenguaje de lista de instrucciones (IL)

El lenguaje de lista de instrucciones o instruction list (IL) es el lenguaje más básico y simple de todos los definidos en la norma IEC 1131-3. Es un lenguaje textual de bajo nivel con sintaxis similar al ensamblador. Un programa IL consiste en una lista de instrucciones compuestas por un operador y por uno o ningún operando, y que pueden estar precedidas o no por una etiqueta seguida de ':'. En cualquier lugar del programa pueden aparecer comentarios delimitados por '(*' y '*)'.

El siguiente es un ejemplo de un programa IL:

LABEL | OPERATOR | OPERAND | COMMENT

```
START:      LD        %I0.0      (* Start pushbutton *)
            AND       %I0.1      (* Enable *)
            ST        %Q0.0      (* Motor on *)
            END
```

El resultado de las operaciones se almacena en dos acumuladores especiales llamados:

- RLO (result of logic operation o resultado de la operación lógica, de 1 bit) para operaciones lógicas
- ACC (*accumulator o acumulador,* de 16 bits) para operaciones aritméticas.

Los operadores IL están sobrecargados, esto es, un mismo operador puede trabajar y hacer cosas distintas en función del tipo de datos del operando. Así, por ejemplo "LD %I0.0" cargaría el RLO con el valor de la entrada I0.0. Por otro parte, "LD %IW0" cargaría en ACC el valor de la primera palabra de entradas, dejando el RLO sin cargar.

El tipo ACC no es fijo, sino que varía en el tiempo. Cualquier operación de carga, fija el tipo del ACC según la variable cargada y cualquier operación siguiente en el ACC puede cambiarlo. Observa el siguiente ejemplo:

```
LD    1          (* ACC tipo es INT *)
LD    1.0        (* ACC tipo es REAL *)

LD    1          (* ACC tipo es INT *)
ADD   1.0    (* El operando es real, por so ACC es convertido a REAL *)
```

7.2. - Tipos de datos

Los tipos de datos IEC soportados son los siguientes:

Tipo IEC	Descripción	Tamaño bits	Valores permitidos
BOOL	Bit	1	TRUE/FALSE
BYTE	Byte sin signo	8	0…255
SINT	Byte con signo	8	-128…127
WORD	Word sin signo	16	0…65535
INT	Word con signo	16	-32768…32767
DWORD	Doble word sin signo	32	0…4294967295
DINT	Doble word con signo	32	2147483648…+2147483647
REAL	Valor de punto flotante (IEEE-754)	32	

7.3. - Instrucciones

En la tabla siguiente se describirán las instrucciones soportadas. Para cada instrucción se detallan los tipos de datos y las aéreas de memoria soportadas, y también el tamaño de la instrucción en bytes. El tamaño de la instrucción depende del tipo de dato de operando, por ello se detallan los tamaños mínimos y máximos. El tamaño máximo de cualquier instrucción es de 6 bytes.

Además, para cada instrucción se detalla si es o no una instrucción estándar IEC. Instrucciones como INC, DEC, CLR… no son estándar, pero se han incorporado porque permiten hacer operaciones muy comunes con una sola instrucción en lugar de varias.

Muchas instrucciones pueden ser usadas con o sin operando. Si se utilizan operandos, estas llevan a cabo las operaciones del operando. Si se usa sin operando, estas realizan la misma operación en el RLO o en el ACC.

[illegible]	[illegible]	[illegible]	Tipo de Datos	Área de Memoria	Descripción

			Bool	Byte	Sint	Word	Int	Dword	Dint	Real	C	I	Q	M	V	S	
Instrucciones de Transferencia																	
LD	●	3-6	●	●	●	●	●	●	●	●	●	●	●	●	●	●	Carga el valor del operando en el RLO (si el tipo es BOOL) o en el ACC (en caso contrario).
LDN	●	3-6	●	●	●	●	●	●	●	●	●	●	●	●	●	●	Carga el valor negado del operando en el RLO (si el tipo es BOOL) o en el ACC (en caso contrario).
ST	●	3-6	●	●	●	●	●	●	●	●		●	●	●	●	●	Almacena el valor del RLO (si el tipo es BOOL) o del ACC (en caso contrario) en el operando...
STN	●	3-6	●	●	●	●	●	●	●	●		●	●	●	●	●	Almacena el valor negado del RLO (si el tipo es BOOL) o del ACC (en caso contrario) en el operando.
S[1]	●	1-4	●									●	●	●	●	●	Pone a TRUE el operando sólo si RLO vale TRUE. Si se utiliza sin operando pone a TRUE el RLO incondicionalmente (Esto es equivalente a LD TRUE
R[1]	●	1-4	●									●	●	●	●	●	Pone a FALSE el operando sólo si RLO vale TRUE. Si se utiliza sin operando pone a FALSE el RLO incondicionalmente (Esto es equivalente a LD FALSE
Instrucciones lógicas																	
NOT[1]	●	1-4	●	●		●		●				●	●	●	●	●	Invierte bit a bit el operando. Si se utiliza sin operando invierte el RLO.
AND	●	3-6	●	●		●		●			●	●	●	●	●	●	Operación lógica AND del RLO (si el tipo es BOOL) o del el ACC (en caso contrario) con el operando.
AND(	●	1															Operación lógica AND del RLO con el resultado del paréntesis[2].
ANDN	●	3-6	●	●		●		●			●	●	●	●	●	●	Operación lógica AND del RLO (si el tipo es BOOL) o del el ACC (en caso contrario) con el operando negado.
ANDN(	●	1															Operación lógica AND negada del RLO con el resultado del paréntesis[2].
OR	●	3-6	●	●		●		●			●	●	●	●	●	●	Operación lógica OR del RLO (si el tipo es BOOL) o del el ACC (en caso contrario) con el operando.
OR(	●	1															Operación lógica OR del RLO con el resultado del paréntesis[2].

r u c . E (b	Tipo de Data	Áreas de Memoria	Descripción

[1] Puede ser usada con o sin operando.

[2] Si se anidan más de 4 paréntesis se producirá un error fatal y se detendrá la ejecución del programa. Lo mismo ocurrirá si se llega al final del programa con algún paréntesis abierto.

			Bool	Byte	Sint	Word	Int	Dword	Dint	Real	C	I	Q	M	V	S	
ORN	●	3-6	●	●		●		●			●	●	●	●	●	●	Operación lógica OR del RLO (si el tipo es BOOL) o del el ACC (en caso contrario) con el operando negado.
ORN(	●	1															Operación lógica OR negada de RLO con el resultado del paréntesis[3].
XOR	●	3-6	●	●		●		●			●	●	●	●	●	●	Operación lógica XOR del RLO (si el tipo es BOOL) o del el ACC (en caso contrario) con el operando.
XOR(	●	1															Operación lógica XOR del RLO con el resultado del paréntesis[3].
XORN	●	3-6	●	●		●		●			●	●	●	●	●	●	Operación lógica XOR del RLO (si el tipo es BOOL) o del el ACC (en caso contrario) con el operando negado.
XORN(	●	1															Operación lógica XOR negada de RLO con el resultado del paréntesis[3].
)	●	1															Final de operaciones entre paréntesis[4].
Instrucciones Aritméticas																	
ADD	●	3-6		●	●	●	●	●	●	●	●	●	●	●	●	●	Suma aritmética del ACC con el operando y guarda el resultado en ACC (ACC = ACC + OP)
SUB	●	3-6		●	●	●	●	●	●	●	●	●	●	●	●	●	Resta aritmética del ACC con el operando y guarda el resultado en ACC (ACC = ACC - OP)
MUL	●	3-6		●	●	●	●	●	●	●	●	●	●	●	●	●	Multiplicación aritmética del ACC con el operando y guarda el resultado en ACC (ACC = ACC * OP)
DIV	●	3-6		●	●	●	●	●	●	●	●	●	●	●	●	●	División aritmética del ACC con el operando y guarda el resultado en ACC (ACC = ACC / OP)[5].
MOD	●	3-6		●	●	●	●	●	●		●	●	●	●	●	●	División aritmética del ACC con el operando y guarda el resto en ACC (ACC = ACC % OP)[5].

[3] Si se anidan más de 4 paréntesis se producirá un error fatal y se detendrá la ejecución del programa. Lo mismo ocurrirá si se llega al final del programa con algún paréntesis abierto.

[4] El Paréntesis debe de estar abierto antes de ser cerrado, se producirá un error fatal y se detendrá la ejecución del programa

[5] Si el operando es 0, se producirá un error fatal y se detendrá la ejecución del programa.

[6] Puede ser utilizado con o sin operando.

Instrucción	IEC	Size (bytes)	Tipo de datos								Área de memorias						Descripción
			Bool	Byte	Sint	Word	Int	Dword	Dint	Real	C	I	Q	M	V	S	
INC[6]		1-4		•	•	•	•	•	•			•	•	•	•	•	Incrementa el operando (OP = OP + 1). Si se usa sin operando se incrementa el ACC (ACC = ACC + 1).
DEC[6]			•	•	•	•	•	•			•	•	•	•	•		Decrementa el operando (OP = OP - 1). Si se usa sin operando decrementa el ACC (ACC = ACC - 1).
CLR[6]			•	•	•	•	•	•			•	•	•	•	•		Pone a 0 el operando (OP = 0). Si se pone si operando, pone ACC a 0 (ACC = 0).
SHL[7]	•	1-4		•	•	•	•	•	•			•	•	•	•	•	Desplaza el operando 1 bit a la izquierda (OP = OP << 1). Si se usa sin operando desplaza el ACC 1 bit a la izquierda (ACC = ACC << 1).
SHR[7]	•	1-4		•	•	•	•	•	•			•	•	•	•	•	Desplaza el operando 1 bit a la derecha (OP = OP >> 1). Si se usa sin operando desplaza el ACC 1 bit a la derecha (ACC = ACC >> 1).
Instrucciones de comparación																	
GT	•	3-6		•	•	•	•	•	•	•	•	•	•	•	•	•	Pone a TRUE el RLO si el ACC es mayor que el operando. En caso contrario lo pone a FALSE
GE	•	3-6		•	•	•	•	•	•	•	•	•	•	•	•	•	Pone a TRUE el RLO si el ACC es mayor o igual que el operando. En caso contrario lo pone a FALSE
EQ	•	3-6		•	•	•	•	•	•	•	•	•	•	•	•	•	Pone a TRUE el RLO si el ACC es igual que el operando. En caso contrario lo pone a FALSE
NE	•	3-6		•	•	•	•	•	•	•	•	•	•	•	•	•	Pone a TRUE el RLO si el ACC es distinto que el operando. En caso contrario lo pone a FALSE
LE	•	3-6		•	•	•	•	•	•	•	•	•	•	•	•	•	Pone a TRUE el RLO si el ACC es menor o igual que el operando. En caso contrario lo pone a FALSE
LT	•	3-6		•	•	•	•	•	•	•	•	•	•	•	•	•	Pone a TRUE el RLO si el ACC es menor que el operando. En caso contrario lo pone a FALSE

[7] Puede ser utilizado con o sin operando.



Instrucción	IEC	Size (bytes)	Tipo de datos								Áreas de Memoria						Descripción
			Bool	Byte	Sint	Word	Int	Dword	Dint	Real	C	I	Q	M	V	S	
Instrucciones de control de programa																	
JMP	●	4				●					●						Salta a la dirección indicada en el operando.
JMPC	●	4				●					●						Salta condicionalmente (cuando RLO es TRUE) a la dirección indicada por el operando.
JMPCN	●	4				●					●						Salta condicionalmente (cuando RLO es FALSE) a la dirección indicada por el operando.
CAL[8]	●	4				●					●						Llama incondicionalmente a la subrutina cuya dirección indica el operando.
CALC[8]	●	4				●					●						Llama condicionalmente (cuando RLO es TRUE) a la subrutina cuya dirección indica el operando.
CALCN[8]	●	4				●					●						Llama condicionalmente (cuando RLO es FALSE) a la subrutina cuya dirección indica el operando.
LOOP		4				●					●						Decrementa ACC and y salta a la dirección indicada por el operando si ACC es 0.
LOOPN		4				●					●						Decrementa ACC and y salta a la dirección indicada por el operando si ACC no es 0.
RET[9]	●	1															Finaliza la subrutina y retorna a la instrucción siguiente a la llamada (CAL).
RETC[9]	●	1															Finaliza la subrutina condicionalmente (cuando RLO es TRUE) y retorna a la instrucción siguiente a la llamada (CAL).
RETCN[9]	●	1															Finaliza la subrutina condicionalmente (cuando RLO es FALSE) y retorna a la instrucción siguiente a la llamada (CAL).
END	●	1															Indica el final del programa de usuario. Su uso es obligatorio.

7 Puede ser usado con o sin operando.
8 Si se anidan más de 4 llamadas, se producirá un error fatal y se detendrá la ejecución del programa.
9 Si es llamada fuera de una subrutina, se producirá un error fatal y se detendrá la ejecución del programa.

Instrucción	IEC	Size (bytes)	Tipo de dato								Área de memoria						Descripción
			Bool	Byte	Sint	Word	Int	Dword	Dint	Real	C	I	Q	M	V	S	
Instrucciones variadas																	
NOP		1															No operación.
RTRIG		1-4	•											•	•		Detecta un flanco positivo del RLO, y guarda el valor anterior en el operando.
FTRIG		1-4	•											•	•		Detecta un flanco negativo del RLO, y guarda el valor anterior en el operando.
Instrucciones de acumulación																	
SWAP		1															Cambia el orden de los bytes del ACC.
ABS		1															Valor absoluto del ACC.
SQRT		1															Raíz cuadrada del ACC.
LN		1															Logaritmo natural del ACC.
LOG		1															Logaritmo en Base-10 ACC.
EXP		1															Exponencial del ACC.
SIN		1															Seno del ACC.
COS		1															Coseno del ACC.
TAN		1															Tangente del ACC.
ASIN		1															Arc seno del ACC.
ACOS		1															Arc coseno del ACC.
ATAN		1															Arc tangente del ACC.
Instrucciones de conversión																	
TO_BYTE		1															Convierte ACC a BYTE.
TO_SINT		1															Convierte ACC a SINT.
TO_WORD		1															Convierte ACC a WORD.
TO_INT		1															Convierte ACC a INT.
TO_DWORD		1															Convierte ACC a DWORD.
TO_DINT		1															Convierte ACC a DINT.
TO_REAL		1															Convierte ACC a REAL.
BCD		1															Convierte ACC de decimal a BCD.
I_BCD		1															Convierte ACC de BCD a decimal.

7.4. – Áreas de memoria

Los operandos que acompañan a las instrucciones pueden ser constantes o pertenecer a una de las siguientes áreas de memoria:

Área	Descripción	Tamaño (words)	Rango
I	Memoria de entradas	9	%IW0…%IW8
Q	Memoria de salidas	1	%QW0
M	Memoria de RAM de usuario (no persistente)	256	%MW0…%MW255
V	EEPROM memoria de usuario (persistente)	256	%VW0…%VW255
S	Memoria de sistema	256	%SW0…%SW255

La lectura de los registros fuera de los valores mostrados en la tabla anterior (por ejemplo %IW9 no causará error, pero siempre retornará 0. La escritura en un registro no valido no provocará ningún cambio.

Como puede observarse en la tabla anterior, instrucciones como ST, S, R, etc… también pueden operar sobre la memoria de entradas, lo cual puede ser muy útil cuando se necesita forzar o simular el funcionamiento de los elementos a controlar.

La zona de sistema de memoria es sólo de lectura. Escribir en sus registros no causará ningún efecto (y tampoco errores). Las otras áreas tienen acceso de lectura y escritura.

La memoria V (no volátil) de trabajo se encuentra realmente en memoria RAM, y se vuelca periódicamente a la EEPROM del microcontrolador. Debido a que el número de escrituras de la EEPROM es limitado, y con el fin de maximizar la vida útil de esta memoria, este volcado se realiza de manera automática una vez por hora. Este largo periodo de tiempo podría provocar la pérdida de datos importantes si se produce una pérdida de alimentación antes de que se realice el volcado. Para evitar esto, se dispone del bit de sistema %S1.8 que puede ser activado desde programa de usuario para forzar un volcado instantáneo a la EEPROM y asegurar los datos críticos.

También para maximizar la duración de esta memoria, el volcado se realiza de forma inteligente, esto es, solamente se actualizan los registros que han sido modificados.

La integridad de la memoria V se chequea en el arranque de la placa y se almacena en el bit de sistema %S0.7. Si este bit está activo, la memoria EEPROM está corrompida y los datos almacenados en memoria V pueden no ser válidos. El programa debería comprobar este bit en el arranque y, si está activo, podría restaurar valores por defecto de la aplicación y avisar al usuario con una alarma.

7.4.1. – Área de entradas

Registro	Descripción
%IW0	Registro de entradas digitales (lectura/escritura): Bit 0: Entrada digital DI0 Bit 1: Entrada digital DI1 Bit 2: Entrada digital DI2 … Bit 15: Entrada digital DI15 Si una entrada se configura como analógica, el correspondiente bit se activará si el valor analógico se encuentra por encima del 50% del máximo valor analógico (5V / 20mA), y no se activará en otros casos.
%IW1…%IW8	Escalado analógico de las entradas AI0…AI7 (lectura/escritura) SINT tipo valor en el rango -32768…32767.

Tabla interna de %IW0:

Bit	Descripción
0	Entrada digital DI0
1	Entrada digital DI1
2	Entrada digital DI2
…	
15	Entrada digital DI15

7.4.2.- Área de salidas

Registro	Descripción
%QW0	Salidas digitales (lectura/escritura):

Bit	Descripción
0	Salida digital DI0
1	Salida digital DI1
2	Salida digital DI2
…	
11	Salida digital DI11
12	No usado
13	No usado
14	No usado
15	No usado

7.4.3.- Área de RAM de usuario

Registro	Descripción
%MW0…%MW255	Zona de memoria de RAM de usuario. Estos registros de usuario son volátiles, con lo que su contenido se perderá al quitar la alimentación.

7.4.4.- Área de EEPROM de usuario

Registro	Descripción
%VW0…%VW255	Zona de memoria de EEPROM de usuario. Estos registros de usuario son persistentes, con lo que su contenido se mantendrá al quitar la alimentación.



7.4.5. - Área de sistema

<table>
<tr><th>Registro</th><th>Descripción</th></tr>
<tr><td>%SW0</td><td>Registro de estado general (sólo lectura):
<table>
<tr><th>Bit</th><th>Descripción</th></tr>
<tr><td>%S0.0</td><td>Bit siempre FALSE.</td></tr>
<tr><td>%S0.1</td><td>Bit siempre TRUE.</td></tr>
<tr><td>%S0.2</td><td>Primer ciclo. TRUE en el primer ciclo del programa después del paso de STOP a RUN, y FALSE en el resto de ciclos.</td></tr>
<tr><td>%S0.3</td><td>No usado.</td></tr>
<tr><td>%S0.4</td><td>Este bit indica si el última reset fue por apagado de alimentación (normal)</td></tr>
<tr><td>%S0.5</td><td>Este bit indica si el última reset fue por desvanecimiento (brown-out) de la alimentación</td></tr>
<tr><td>%S0.6</td><td>Este bit indica si el última reset fue por bloqueo (watchdog) del microcontrolador de la placa</td></tr>
<tr><td>%S0.7</td><td>Este bit indica un error de checksum de la EEPROM. El contenido de la memoria no volátil se comprueba al arrancar, y si no es correcto se activa este bit. El programa de usuario debería usar este bit para cargar los valores por defecto de las variables importantes del programa.</td></tr>
<tr><td>%S0.8</td><td>Este bit indica un error de checksum de la memoria de programa. El contenido de la memoria de programa se comprueba antes del paso a RUN, y si no es correcto se activa este bit y se pasa a STOP.</td></tr>
<tr><td>%S0.9</td><td>Este bit indica si hay cambios en memoria no volátil (V) pendientes de ser escritos a la EEPROM. La memoria V se vuelca periódicamente a la EEPROM, pero el programa de usuario puede, monitorizar este bit, forzar un volcado instantáneo para asegurar el volcado de datos críticos.</td></tr>
<tr><td>%S0.10</td><td>Este bit está activo cuando el programa de usuario está en ejecución (modo RUN), y está inactivo en modo STOP.</td></tr>
<tr><td>%S0.11</td><td>Reservado.</td></tr>
<tr><td>%S0.12</td><td>Reservado.</td></tr>
<tr><td>%S0.13</td><td>No usado.</td></tr>
<tr><td>%S0.14</td><td>No usado.</td></tr>
<tr><td>%S0.15</td><td>No usado.</td></tr>
</table>
</td></tr>
<tr><td>%SW1</td><td>Registro de comandos generales (lectura/escritura):
<table>
<tr><th>Bit</th><th>Descripción</th></tr>
<tr><td>%S1.0</td><td>Reset de fallos.</td></tr>
<tr><td>%S1.1</td><td>Reset microcontrolador (software reset).</td></tr>
<tr><td>%S1.2</td><td>Reset máximo del tiempo de ciclo máximo.</td></tr>
<tr><td>%S1.3</td><td>Comienzo de la ejecución del programa (RUN modo Petición).</td></tr>
<tr><td>%S1.4</td><td>No usado.</td></tr>
<tr><td>%S1.5</td><td>Reservado.</td></tr>
<tr><td>%S1.6</td><td>Reservado.</td></tr>
<tr><td>%S1.7</td><td>Habilitar el forzado de entradas y salidas (ver %SW20…%SW23)</td></tr>
<tr><td>%S1.8</td><td>Forzado del volcado de memoria V a la EEPROM. El volcado no se realizará inmediatamente después de forzar el bit, pero si al final del ciclo de scan del PLC.</td></tr>
<tr><td>%S1.8</td><td>No usado.</td></tr>
<tr><td>%S1.9</td><td>No usado.</td></tr>
<tr><td>%S1.10</td><td>No usado.</td></tr>
<tr><td>%S1.11</td><td>No usado.</td></tr>
<tr><td>%S1.12</td><td>No usado.</td></tr>
<tr><td>%S1.13</td><td>No usado.</td></tr>
<tr><td>%S1.14</td><td>No usado.</td></tr>
<tr><td>%S1.15</td><td>No usado.</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Registro</th><th>Descripción</th></tr>
<tr><td>%SW2</td><td>Registro de flags temporizados (sólo lectura):
<table>
<tr><th>Bit</th><th>Descripción</th></tr>
<tr><td>%S2.0</td><td>Flag activo cada 100 ms.</td></tr>
<tr><td>%S2.1</td><td>Flag activo cada 200 ms.</td></tr>
<tr><td>%S2.2</td><td>Flag activo cada 500 ms.</td></tr>
<tr><td>%S2.3</td><td>Flag activo cada 1 segundo.</td></tr>
<tr><td>%S2.4</td><td>Flag activo cada 2 segundos.</td></tr>
<tr><td>%S2.5</td><td>Flag activo cada 5 segundos.</td></tr>
<tr><td>%S2.6</td><td>Flag activo cada 10 segundos.</td></tr>
<tr><td>%S2.7</td><td>Flag activo cada 15 segundos.</td></tr>
<tr><td>%S2.8</td><td>Flag activo cada 30 segundos.</td></tr>
<tr><td>%S2.9</td><td>Flag activo cada 1 minuto.</td></tr>
<tr><td>%S2.10</td><td>Flag activo cada 1 hora.</td></tr>
<tr><td>%S2.11</td><td>Flag activo cada 1 dia.</td></tr>
<tr><td>%S2.12</td><td>No usado.</td></tr>
<tr><td>%S2.13</td><td>No usado.</td></tr>
<tr><td>%S2.14</td><td>No usado.</td></tr>
<tr><td>%S2.15</td><td>No usado.</td></tr>
</table></td></tr>
<tr><td>%SW3</td><td>Registro de pulsos temporizados (sólo lectura):
<table>
<tr><th>Bit</th><th>Descripción</th></tr>
<tr><td>%S3.0</td><td>Pulso con 100 ms periodo (10 Hz).</td></tr>
<tr><td>%S3.1</td><td>Pulso con 200 ms periodo (5 Hz).</td></tr>
<tr><td>%S3.2</td><td>Pulso con 500 ms periodo (2 Hz).</td></tr>
<tr><td>%S3.3</td><td>Pulso con 1 segundo de periodo (1 Hz).</td></tr>
<tr><td>%S3.4</td><td>Pulso con 2 segundos de periodo (0.5 Hz).</td></tr>
<tr><td>%S3.5</td><td>Pulso con 5 segundos de periodo (0.2 Hz).</td></tr>
<tr><td>%S3.6</td><td>Pulso con 10 segundos de periodo (0.1 Hz).</td></tr>
<tr><td>%S3.7</td><td>Pulso con 15 segundos de periodo.</td></tr>
<tr><td>%S3.8</td><td>Pulso con 30 segundos de periodo.</td></tr>
<tr><td>%S3.9</td><td>Pulso con 1 minute de periodo.</td></tr>
<tr><td>%S3.10</td><td>Pulso con 1 hora de periodo.</td></tr>
<tr><td>%S3.11</td><td>Pulso con 1 día de periodo.</td></tr>
<tr><td>%S3.12</td><td>No usado.</td></tr>
<tr><td>%S3.13</td><td>No usado.</td></tr>
<tr><td>%S3.14</td><td>No usado.</td></tr>
<tr><td>%S3.15</td><td>No usado.</td></tr>
</table></td></tr>
<tr><td>%SW4</td><td>Contador de milisegundos (sólo lectura). Comienza en 0 cuando se arranca la placa, e incrementa cada milisegundo, volviendo a 0 tras llegar a 65535 ciclos.
<table>
<tr><th>Bit</th><th>Descripción</th></tr>
<tr><td>%S4.0</td><td>Pulso con 1 ms de periodo (1000 Hz).</td></tr>
<tr><td>%S4.1</td><td>Pulso con 2 ms de periodo (500 Hz).</td></tr>
<tr><td>%S4.2</td><td>Pulso con 4 ms de periodo (250 Hz).</td></tr>
<tr><td>%S4.3</td><td>Pulso con 8 ms de periodo (125 Hz).</td></tr>
<tr><td colspan="2">…</td></tr>
<tr><td>%S4.14</td><td>Pulso con 16384 ms de periodo</td></tr>
<tr><td>%S4.15</td><td>Pulso con 32768 ms de periodo</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Registro</th><th>Descripción</th></tr>
<tr><td>%SW5</td><td>Contador de ciclos. Este registro comienza a 0 cuando arranca la placa y se incrementa en cada ciclo de scan, volviendo a 0 tras llegar a 65535 ciclos.</td></tr>
<tr><td>%SW6</td><td>Configuración de interruptores (sólo lectura). En el arranque, la configuración de los interruptores se mapea al registro (ver sección 5)
<table>
<tr><th>Bit</th><th>Descripción</th></tr>
<tr><td>%S6.0</td><td>Dirección Modbus bit 0 (ver sección 5.1)</td></tr>
<tr><td>%S6.1</td><td>Dirección de Modbus bit 1.</td></tr>
<tr><td>%S6.2</td><td>Dirección de Modbus bit 2.</td></tr>
<tr><td>%S6.3</td><td>Dirección de Modbus bit 3</td></tr>
<tr><td>%S6.4</td><td>Dirección de Modbus bit 4.</td></tr>
<tr><td>%S6.5</td><td>Dirección de Modbus bit 5.</td></tr>
<tr><td>%S6.6</td><td>Dirección de Modbus bit 6.</td></tr>
<tr><td>%S6.7</td><td>Dirección de Modbus bit 7.</td></tr>
<tr><td>%S6.8</td><td>Velocidad transmisión S1 (ver sección5.2).</td></tr>
<tr><td>%S6.9</td><td>Velocidad transmisión S2.</td></tr>
<tr><td>%S6.10</td><td>Velocidad transmisión S3.</td></tr>
<tr><td>%S6.11</td><td>Interruptor de usuario (ver sección 5.3).</td></tr>
<tr><td>%S6.12</td><td>No usado.</td></tr>
<tr><td>%S6.13</td><td>No usado.</td></tr>
<tr><td>%S6.14</td><td>No usado.</td></tr>
<tr><td>%S6.15</td><td>No usado.</td></tr>
</table></td></tr>
<tr><td>%SW7</td><td>Último tiempo de ciclo en milisegundos (sólo lectura).</td></tr>
<tr><td>%SW8</td><td>Tiempo de ciclo máximo en milisegundos (sólo lectura). Reset activando %S1.2.</td></tr>
<tr><td>%SW9…%SW10</td><td>Reservado.</td></tr>
<tr><td>%SW11</td><td>Contador de programa (sólo lectura).</td></tr>
<tr><td>%SW12</td><td>Código de error de programa (sólo lectura, ver sección 7.5).</td></tr>
<tr><td>%SW13</td><td>Contador de instrucciones de programa con error (sólo lectura).</td></tr>
<tr><td>%SW14</td><td>Contador del número de llamadas anidadas (sólo lectura).</td></tr>
<tr><td>%SW15</td><td>Contador del número de paréntesis anidados (sólo lectura).</td></tr>
<tr><td>%SW16…%SW19</td><td>Reservado.</td></tr>
<tr><td>%SW20</td><td>Mascara de forzado a OFF de las entradas (lectura/escritura).Cuando el forzado de I/O está activo (%S1.7), cada bit en este registro provocará el forzado a OFF de las correspondientes entradas, independientemente del voltaje del conector físico.</td></tr>
<tr><td>%SW21</td><td>Mascara de forzado a OFF de las entradas (lectura/escritura).Cuando el forzado de I/O está activo (%S1.7), cada bit en este registro provocará el forzado a ON de las correspondientes entradas, independientemente del voltaje del conector físico.</td></tr>
<tr><td>%SW22</td><td>Mascara de forzado a OFF de las salidas (lectura/escritura).Cuando el forzado de I/O está activo (%S1.7), cada bit en este registro provocará el forzado a OFF de las correspondientes salidas, independientemente del voltaje del conector físico.</td></tr>
<tr><td>%SW23</td><td>Mascara de forzado a OFF de las salidas (lectura/escritura).Cuando el forzado de I/O está activo (%S1.7), cada bit en este registro provocará el forzado a ON de las correspondientes salidas, independientemente del voltaje del conector físico.</td></tr>
<tr><td>%SW30…%SW37</td><td>Entrada analógica sin escalado AI0…AI7 (sólo lectura).
Tipo WORD de rango 0…1023.</td></tr>
</table>

<table>
<tr><th>Registro</th><th>Descripción</th></tr>
<tr><td>%SW40…%SW44</td><td>Punteros de direccionamiento indirecto a memoria M (lectura/escritura).
Estos punteros permiten direccionar la memoria M a través de los registros virtuales %SW50-%SW54. Por ejemplo, si cargamos el valor 10 en el registro %SW40, cuando accedamos al registro virtual %SW50 realmente estaremos accediendo al registro real %MW10.</td></tr>
<tr><td>%SW45…%SW49</td><td>Punteros de direccionamiento indirecto a memoria V (lectura/escritura).
Estos punteros permiten direccionar la memoria V a través de los registros virtuales %SW55-%SW59. Por ejemplo, si cargamos el valor 10 en el registro %SW45, cuando accedamos al registro virtual %SW55 realmente estaremos accediendo al registro real %VW10. Esto es muy útil para operaciones como bucles.</td></tr>
<tr><td>%SW50…%SW54</td><td>Registros virtuales de direccionamiento indirecto a memoria M (lectura/escritura).</td></tr>
<tr><td>%SW55…%SW59</td><td>Registros virtuales de direccionamiento indirecto a memoria V (lectura/escritura).</td></tr>
<tr><td>%SW60</td><td>Registro de estado de comunicaciones serie (sólo lectura):
<table>
<tr><th>Bit</th><th>Descripción</th></tr>
<tr><td>%S60.0</td><td>Error de timeout de recepción (RX)</td></tr>
<tr><td>%S60.1</td><td>Error de desbordamiento del Buffer (overrun)</td></tr>
<tr><td>%S60.2</td><td>Error de formato (framing)</td></tr>
<tr><td>%S60.3</td><td>Error de paridad (parity error).</td></tr>
<tr><td>%S60.4</td><td>Error Buffer lleno.</td></tr>
<tr><td>%S60.5</td><td>No usado.</td></tr>
<tr><td>…</td><td></td></tr>
<tr><td>%S60.15</td><td>No usado.</td></tr>
</table></td></tr>
<tr><td>%SW61</td><td>Registro de comandos de comunicaciones serie (lectura/escritura):
<table>
<tr><th>Bit</th><th>Descripción</th></tr>
<tr><td>%S61.0</td><td>Reseteo de errores en el Puerto serie mediante %SW60.</td></tr>
<tr><td>%S61.1</td><td>No usado.</td></tr>
<tr><td>…</td><td></td></tr>
<tr><td>%S61.15</td><td>No usado.</td></tr>
</table></td></tr>
<tr><td>%SW62</td><td>Velocidad de transmisión del puerto serie (lectura/escritura). En el arranque, esta configuración se lee del banco de interruptores DIP (ver sección 5.2), pero puede sobrescribirse por programa en cualquier momento. Los posibles valores son:
<table>
<tr><th>Valor</th><th>Velocidad de comunicación (bps)</th></tr>
<tr><td>0</td><td>4800</td></tr>
<tr><td>1</td><td>9600</td></tr>
<tr><td>2</td><td>19200</td></tr>
<tr><td>3</td><td>28800</td></tr>
<tr><td>4</td><td>38400</td></tr>
<tr><td>5</td><td>57600</td></tr>
<tr><td>6</td><td>115200</td></tr>
<tr><td>7</td><td>128000</td></tr>
</table></td></tr>
<tr><td>%SW63</td><td>Puerto serie en milisegundos (lectura/escritura).</td></tr>
<tr><td>%SW64</td><td>Bytes recibidos por el puerto serie (sólo lectura).Indica el numero de bytes recibidos en el buffer del puerto serie.</td></tr>
</table>

<table>
<tr><th>Registro</th><th>Descripción</th></tr>
<tr><td>%SW70</td><td>Registro de estado de comunicaciones modbus (sólo lectura):
<table>
<tr><th>Bit</th><th>Descripción</th></tr>
<tr><td>%S70.0</td><td>Error de link. Este bit se activa cuando no se detectan comunicaciones entre la placa y el maestro modbus durante un tiempo (modbus link timeout, ver sección 6.8.1).</td></tr>
<tr><td>%S70.1</td><td>Error de control de trama.</td></tr>
<tr><td>%S70.2</td><td>Error de longitud de trama.</td></tr>
<tr><td>%S70.3</td><td>Error de checksum.</td></tr>
<tr><td>%S70.4</td><td>Error de timeout.</td></tr>
<tr><td>%S70.5</td><td>No usado.</td></tr>
<tr><td colspan="2">…</td></tr>
<tr><td>%S70.15</td><td>No usado.</td></tr>
</table></td></tr>
<tr><td>%SW71</td><td>Registro de comandos de modbus (lectura/escritura):
<table>
<tr><th>Bit</th><th>Descripción</th></tr>
<tr><td>%S71.0</td><td>Reset de errors de modbus en %SW70.</td></tr>
<tr><td>%S71.1</td><td>Reset contadores de diagnóstico de modbus.</td></tr>
<tr><td>%S71.2</td><td>No usado.</td></tr>
<tr><td colspan="2">…</td></tr>
<tr><td>%S71.15</td><td>No usado.</td></tr>
</table></td></tr>
<tr><td>%SW72</td><td>Dirección de esclavo modbus (lectura/escritura). En el arranque, se lee la configuración desde el banco de interruptores DIP (ver sección 5.1), pero puede sobrescribirse por programa en cualquier momento.
Los Valores válidos son 1…246.</td></tr>
<tr><td>%SW73</td><td>Retraso inicial de modbus (lectura/escritura).En el arranque, la placa espera este tiempo (en milisegundos) antes de iniciar las comunicaciones.</td></tr>
<tr><td>%SW74</td><td>Retraso de respuesta del modbus (lectura/escritura). Después de recibir la petición de modbus, la placa espera este tiempo (en milisegundos) antes de enviar la respuesta.</td></tr>
<tr><td>%SW75</td><td>Valor de identificador de la placa (sólo lectura).Este valor identifica el modelo de la placa (7001 para este modelo).</td></tr>
<tr><td>%SW76</td><td>Valor de identificación de hardware de la placa (sólo lectura)</td></tr>
<tr><td>%SW77</td><td>Valor de identificación de firmware de la placa (sólo lectura)</td></tr>
<tr><td>%SW80…%SW81</td><td>Contador de diagnostico de modbus para tramas detectadas en el bus (sólo lectura).
Es un contador de 32 bits (%SW80 high word; %SW81 low word).
Puede resetarse activando el bit %S71.1</td></tr>
<tr><td>%SW82…%SW83</td><td>Contador de diagnostico de modbus para tramas direccionadas al esclavo (sólo lectura).
Es un contador de 32 bits (%SW82 high word; %SW83 low word).
Puede resetarse activando el bit %S71.1</td></tr>
<tr><td>%SW84…%SW85</td><td>Contador de diagnostico de modbus para tramas incompletas direccionadas al esclavo (sólo lectura).
Es un contador de 32 bits (%SW84 high word; %SW85 low word).
Puede resetarse activando el bit %S71.1</td></tr>
<tr><td>%SW86…%SW87</td><td>Contador de diagnostico de modbus para tramas direccionadas al esclavo con errores de checksum (sólo lectura).
Es un contador de 32 bits (%SW86 high word; %SW87 low word).
Puede resetarse activando el bit %S71.1</td></tr>
</table>

<table>
<tr><th>Registro</th><th>Descripción</th></tr>
<tr><td>%SW88…%SW89</td><td>Contador de diagnostico de modbus para tramas direccionadas al esclavo con errores de parámetros (sólo lectura).
Es un contador de 32 bits (%SW88 high word; %SW89 low word).
Puede resetarse activando el bit %S71.1</td></tr>
<tr><td>%SW90…%SW91</td><td>Contador de diagnostico de modbus para tramas direccionadas al esclavo no soportadas (sólo lectura).
Es un contador de 32 bits (%SW90 high word; %SW91low word).
Puede resetarse activando el bit %S71.1</td></tr>
<tr><td>%SW92…%SW93</td><td>Contador de diagnostico de modbus para tramas direccionadas por el esclavo (sólo lectura).
Es un contador de 32 bits (%SW92 high word; %SW93 low word).
Puede resetarse activando el bit %S71.1</td></tr>
<tr><td>%SW100</td><td>Código de error del led (sólo lectura):
<table><tr><th>Valor</th><th>Descripción</th></tr><tr><td>0</td><td>CPU en modo STOP (sin errors)</td></tr><tr><td>1</td><td>CPU en modo RUN (sin errors ni warnings)</td></tr><tr><td>2</td><td>CPU en modo RUN (con warnings)</td></tr><tr><td>3</td><td>CPU en modo STOP (con errors)</td></tr></table></td></tr>
<tr><td>%SW110</td><td>Registro del estado de las variables de tiempo (Time flags, sólo lectura) Este registro es el mismo que el %SW2, pero funciona asincrónicamente con el ciclo de scan principal.</td></tr>
<tr><td>%SW111</td><td>Registro de comandos de tiempo (lectura/escritura):
<table><tr><th>Bit</th><th>Descripción</th></tr><tr><td>%S111.0</td><td>Reset contadores de tiempo en funcionamiento.</td></tr><tr><td>%S111.1</td><td>Reservado.</td></tr><tr><td>%S111.2</td><td>No usado.</td></tr><tr><td>…</td><td></td></tr><tr><td>%S111.15</td><td>No usado.</td></tr></table></td></tr>
<tr><td>%SW112</td><td>Contador de tiempo de la placa que incrementa cada 100 milisegundos de 0-9 (sólo lectura). Puede resetearse activando el bit %S111.0.</td></tr>
<tr><td>%SW113</td><td>Contador de tiempo de la placa que incrementa cada segundos de 0-59 (sólo lectura). Puede resetearse activando el bit %S111.0.</td></tr>
<tr><td>%SW114</td><td>Contador de tiempo de la placa que incrementa cada minuto de 0-59 (sólo lectura). Puede resetearse activando el bit %S111.0.</td></tr>
<tr><td>%SW115</td><td>Contador de tiempo de la placa que incrementa cada hora de 0-23 (sólo lectura). Puede resetearse activando el bit %S111.0.</td></tr>
<tr><td>%SW116</td><td>Contador de tiempo de la placa que incrementa cada día (sólo lectura). Puede resetearse activando el bit %S111.0.</td></tr>
<tr><td>%SW120…%SW127</td><td>Reservado.</td></tr>
<tr><td>%SW130…%SW137</td><td>Límite inferior de escalado par alas entradas analógicas AI0…AI7 (lectura/escritura). Ver sección 6.8.3.</td></tr>
<tr><td>%SW140…%SW147</td><td>Límite superior de escalado para las entradas analógicas AI0…AI7 (lectura/escritura). Ver sección 6.8.3.</td></tr>
<tr><td>%SW150</td><td>Modbus link timeout en milisegundos (lectura/escritura). Ver sección 6.8.1.</td></tr>
</table>

7.5.- Errores y warnings de programa

Existen varias situaciones en las cuales el programa de usuario puede que no se ejecute correctamente. Esta situación será visualizada mediante un parpadeo diferente del led de estado de la placa (ver sección 4.1), pero para una descripción detallada del error es necesario conectarse online y abrir la ventana de estado del dispositivo (ver sección 6.7).

Todos los posibles errores y warnings se encuentran definidos mediante un código numérico que es mapeado al registro de sistema %SW12. El valor 0 indica que no hay error o warning. Cualquier valor entre 1 y 99 indica warning, y cualquier valor de 100 o superior indica error.

Los posibles valores y causas de error son las siguientes:

Cod.	Descripción
0	Ejecución de programa OK (sin error; sin warning)
1	Programa vacio. Esto indica que la placa está intentando pasa a RUN sin un programa descargado.
2	Conversión de tipo con posible información perdida
3	Invalido tipo de acumulador para la intrucción.
4	Operación matemática no posible.
5	División por cero. Si cualquier DIV o MOD se ejecuta cuando el ACC es cero, este warning aparecerá.
100	Tiempo máximo excedido (watchdog). Si el tiempo de ejecución del programa es mayor que tiempo especificado en el watchdog timeout (ver sección 6.8.1) la ejecución se para y aparece este error.
101	Error de checksum de programa. La placa no puede conmutar a modo RUN porque se detectan errores en el programa descargado. El programa debería descargarse otra ver para corregir este error.
102	Código de operación no valida. Todas la posibles instrucciones (ver sección 7.3) tienen un código numérico. Si la placa detecta un código no conocido, esta parará la ejecución y mostrará este error. Normalmente esto es chequeado por el compilador, por eso este error normalmente indica un problema de la memoria del programa.
103	Operador de área no valido. No todos los operadores de área son validos para todas las instrucciones. Por ejemplo, RTRIG o FTRIG son instrucciones que sólo pueden operar con memorias M y V. Si se encuentra alguna combinación inválida, se para la ejecución del programa. Normalmente se chequea mediante el compilador, por eso este error normalmente indica un error de la memoria del programa.
104	Operador de tipo no valido. No todos los tipos de operador son válidos para todas las instrucciones. Por ejemplo, las instrucciones S o R sólo pueden operar con BOOLs. Si se encuentra alguna combinación inválida, se para la ejecución del programa. Normalmente se chequea mediante el compilador, por eso este error normalmente indica un error de la memoria del programa.
105	Paréntesis cerrados o no abiertos. La instrucción ')' debe de utilizarse solamente después de un paréntesis abierto con 'AND(','OR(' o 'XOR('. En cualquier otro caso este error parará la ejecución del programa.

106	Demasiados paréntesis abiertos. No pueden ser anidados más de 4 paréntesis.
107	Demasiadas subrutinas anidadas. No pueden ser anidadas más de 4 CALs.
108	Salto a una dirección incorrecta.
109	RET sin CAL.
110	Final del programa con paréntesis abiertos. Todos los paréntesis abiertos han de ser cerrados antes del final de programa.
111	Final del programa dentro de una subrutina. La instrucción END ha de ser solamente utilizada en el nivel del programa principal.
112	Instrucción no permitida dentro del paréntesis. Las instrucciones de control de programa (JMP, CAL, RET, LOOP) no pueden utilizarse dentro de un paréntesis.



7.6.- Ejemplos de programación

7.6.1.- Puerta automática simple

El siguiente programa de ejemplo serviría para controlar una persiana motorizada con dos pulsadores para subir y bajar, y dos finales de carrera para arriba y abajo:

```
VAR_GLOBAL

      (* Entradas *)
      PB_SUBIR AT %I0.0: BOOL;       (* Pulsador de subir *)
      PB_BAJAR AT %I0.1: BOOL;       (* Pulsador de bajar *)
      SWITCH_ARRIBA AT %I0.2: BOOL; (* Final de carrera arriba *)
      SWITCH_ABAJO AT %I0.3: BOOL;  (* Final de carrera abajo *)

      (* Salidas *)
      SUBIR AT %Q0.0: BOOL;          (* Comando de subir *)
      BAJAR AT %Q0.1: BOOL;          (* Comando de bajar *)

END_VAR

PROGRAM Main

      LD    PB_SUBIR
      ANDN  SWITCH_ARRIBA
      ST    SUBIR

      LD    PB_BAJAR
      ANDN  SWITCH_ABAJO
      ANDN  SUBIR
      ST    BAJAR

      END

END_PROGRAM
```

7.6.2.- Control remoto y local de persiana motorizada

Ahora la persiana motorizada dispone de un control local mediante pulsadores, y un control remoto como un PLC o software SCADA. Un selector de dos posiciones conectado a una entrada selecciona el modo de control local o remoto.

```
VAR_GLOBAL

      (* Entradas *)
      PB_SUBIR AT %I0.0: BOOL;      (* Pulsador de subida *)
      PB_BAJAR AT %I0.1: BOOL;      (* Pulsador de bajada *)
      SWITCH_ARRIBA AT %I0.2: BOOL; (* Final de carrera arriba *)
      SWITCH_ABAJO AT %I0.3: BOOL;  (* Final de carrera abajo *)
      LOCAL AT %I0.4: BOOL;         (* Selector de control Local/Remoto *)

      (* Comandos remotos *)
      REMOTE_SUBIR AT %I0.0: BOOL;  (* Comando remoto de subir *)
      REMOTE_BAJAR AT %I0.1: BOOL;  (* Comando remoto de bajar *)

      (* Salidas *)
      SUBIR AT %Q0.0: BOOL;         (* Comando de Subir *)
      BAJAR AT %Q0.1: BOOL;         (* Comando de bajar *)

END_VAR

PROGRAM Main

      LD    LOCAL
      AND   PB_SUBIR
      OR(
            LDN   LOCAL
            AND   REMOTE_SUBIR
      )
      ANDN  SWITCH_ARRIBA
      ST    SUBIR

      LD    LOCAL
      AND   PB_BAJAR
      OR(
            LDN   LOCAL
            AND   REMOTE_BAJAR
      )
      ANDN  SWITCH_ABAJO
      ANDN  SUBIR
      ST    BAJAR

      END

END_PROGRAM
```

En lugar del selector de modo local/remoto, podría utilizarse el bit que indica el estado el link (%S70.0), de forma que si las comunicaciones están activas la persiana se controla mediante las órdenes remotas, mientras que si se pierde la comunicación con el maestro se conmutaría automáticamente a modo local.

7.6.3.- Temporizaciones

No hay instrucciones para trabajar directamente con temporizadores, pero son muy fáciles de programar utilizando las variables de tiempo situadas en %SW2. Por ejemplo, el código siguiente activa la salida /Q0.0 cuando la entrada %I0.0 está activa durante 5 segundos.

```
VAR_GLOBAL

      IN0 AT %I0.0: BOOL;
      OUT0 AT %Q0.0: BOOL;
      PULSO_1S AT %S2.3: BOOL;
      TIMER AT %MW0: WORD;

END_VAR

PROGRAM Main

            LD    IN0          (* Chequea la entrada *)
            JMPC  CUENTA       (* Si es 1, miramos pulso de 1s *)
            CLR   TIMER        (* Si es 0, reseteamos timer *)
            JMP   CHECK

CUENTA:     LDN   PULSO_1S     (* Pulso de 1 segundo *)
            JMPC  FIN          (* Si es 0, no incrementamos el timer *)
            INC   TIMER        (* Si es 1 , incrementamos el timer *)

FIN:        LD    TIMER              (* Si el timer... *)
            GE    5            (* ...es mayor de 5 segundos... *)
            ST    OUT0         (* ...activamos la salida *)

            END

END_PROGRAM
```

7.6.4.- Intermitencias

Hay varios bits de sistema que pueden ser utilizados para intermitencias constantes, como los mostrados en este ejemplo:

```
PROGRAM Main

      LD    %S3.0       (* Intermitencia 10 Hz *)
      ST    %Q0.0

      LD    %S3.3       (* Intermitencia 1 Hz *)
      ST    %Q0.1

      LD    %S3.9       (* Intermitencia 1/60 Hz *)
      ST    %Q0.2

      LD    %S4.8       (* Intermitencia cada 256ms (4 Hz approx.) *)
      ST    %Q0.2

      LD    %S4.9       (* Intermitencia cada 512ms (2 Hz approx.) *)
      ST    %Q0.2

      LD    %S4.11      (* Intermitencia cada 2048ms (0.5 Hz approx.) *)
      ST    %Q0.2

END_PROGRAM
```

Si necesita de una intermitencia irregular, es necesario trabajar con contadores, como lo mostrado en este ejemplo en el que se activa una salida cada 10 segundos y se desactiva cada 5 segundos

```
VAR_GLOBAL

      OUT0 AT %Q0.0: BOOL;
      PULSO_1S AT %S2.3: BOOL;
      COUNT AT %MW0: WORD;

END_VAR

PROGRAM Main

            LDN   PULSE_1S    (* Pulso de 1 segundo *)
            JMPC  LABEL1
            INC   COUNT       (* Incrementa el contador *)

LABEL1:     LD    TRUE        (* Ponemos RLO a 1 *)
            LD    COUNT       (* Si el timer_. *)
            LT    15          (* Es menor que 15 *)
            JMPC  LABEL2
            CLR   COUNT       (* Resetea si es mayor que 15 segundos (10+5) *)

LABEL2:     LD    TRUE        (* Ponemos RLO a 1 *)
            LD    COUNT
            LT    10
            ST    OUT0        (* Activa la salida si en menor que 10 *)

END_PROGRAM
```

7.6.5.- Gestión de checksum de EEPROM

El siguiente programa efectuaría un reset de los parámetros por defecto si detecta un error de checksum de EEPROM en el arranque del programa:

```
VAR_GLOBAL

      (* Definición de bits de sistema *)
      FIRST_CYCLE AT %S0.2: BOOL;
      EEPROM_ERROR AT %S0.7: BOOL;
      EEPROM_WRITE AT %S1.8: BOOL;
      EEPROM_RESTORED AT %M0.0: BOOL;

      (* Variables en la memoria V *)
      VAR1: INT;
      VAR2: INT;
      VAR3: INT;
      ...

END_VAR

PROGRAM Main

            LD    FIRST_CYCLE        (* Primer ciclo *)
            AND   EEPROM_ERROR       (* Si hay error de checksum de EEPROM... *)
            JMPC  RESET_EE           (* ...se salta a restaurar valores *)
            JMP   MAIN_PROG          (* ...si no, salta a programa principal *)

RESET_EE:   LD    100                (* Valor por defecto para VAR1 *)
            ST    VAR1
            LD    200                (* Valor por defecto para VAR2 *)
            ST    VAR2
            LD    300                (* Valor por defecto para VAR3 *)
            ST    VAR3
            ...
            LD    TRUE
            S     EEPROM_WRITE       (* Forzamos volcado de V memory a EEPROM *)
            S     EEPROM_RESTORED    (* Activamos alarma *)

MAIN_PROG:  LD    TRUE               (* Comienzo del programa principal *)
            ...
            END

END_PROGRAM
```

El bit EEPROM_RESTORED (%M0.0) en el ejemplo se activará para notificar de la alarma a un nivel de sistema superior que informará al usuario que la memoria esta corrupta, y los valores por defecto han sido restaurados. El operador debe de chequear si los valores son los correctos o han cambiado.

En algunos casos, el programa principal no debe de ejecutarse hasta que los parámetros sean correctamente definidos por el usuario. Esto podría ser un ejemplo de ello:

```
VAR_GLOBAL

      (* Bits de sistema *)
      EEPROM_ERROR AT %S0.7: BOOL;
      EEPROM_RESTORED AT %M0.0: BOOL;

END_VAR

PROGRAM Main

            LDN   EEPROM_ERROR
            OR    EEPROM_RESTORED
            JMPC  MAIN_PROG
            END

MAIN_PROG:  LD    TRUE              (* Comienzo del programa principal *)
            ...
            END

END_PROGRAM
```

Este programa se quedaría esperando que el maestro modbus de comunicaciones restaurara los parámetros correctos y tras ello activara el bit EEPROM_RESTORED (%M0.0). A partir de ese momento se ejecutaría el programa principal.

7.6.6.- Uso de los registros de direccionamiento indirecto

El siguiente programa efectúa una copia de los primeros 50 registros de la memoria M a la memoria V haciendo uso de los registros de direccionamiento indirecto y las instrucciones LOOP:

```
VAR_GLOBAL

      POINTER_M AT %SW40: WORD;
      POINTER_v AT %SW45: WORD;
      IND_M AT %SW50: WORD;
      IND_v AT %SW55: WORD;

END_VAR

PROGRAM Main

            LD    0            (* ACC = 0 *)
            ST    POINTER_M    (* Puntero a memoria M *)
            ST    POINTER_V    (* Puntero a memoria V *)

COPY_LOOP:  LD    IND_M        (* ACC = %MW[POINTER_M] *)
            ST    IND_V        (* %VW[POINTER_V] = ACC *)
            INC   POINTER_M    (* Incrementa el puntero a M *)
            INC   POINTER_V    (* Incrementa el puntero a V *)
            LD    POINTER_M
            LE    50           (* Si el puntero es menor o igual 50... *)
            JMPC  COPY_LOOP    (* ...repetimos bucle *)

MAIN_PROG:  LD    TRUE         (* Comienzo del programa principal *)
            ...
            END

END_PROGRAM
```

8.- MODBUS COMMUNICATION

8.1.- Protocolo modbus

El protocolo modbus es un protocolo maestro-esclavo que funciona sobre un bus RS-485. En el bus existe un sólo dispositivo (maestro) que inicia las transacciones modbus, cuyo objeto es leer o escribir datos de los otros dispositivos (esclavos). Los dispositivos esclavos responden al maestro con los datos solicitados, o con respuestas de confirmación o de error.

Se ha escogido este protocolo de comunicaciones entre muchos otros por tratarse de un protocolo de uso muy extendido en entornos industriales pero sobretodo por ser un protocolo sencillo, bien documentado y de uso libre. Se recomienda consultar las especificaciones del protocolo modbus que pueden encontrarse en **www.modbus.org**.

Las tramas de comunicaciones están limitadas a 256 bytes, tal y como recomienda la norma modbus. Esto permite leer o escribir más de 100 registros de 16 bits con una sola trama.

Las comunicaciones serie han de configurarse con 8 bits de datos, 1 bit de stop y sin paridad. La velocidad de comunicaciones puede ser de 4800, 9600, 19200, 2880, 38400, 57600, 115200 O 128000 bps.

En caso de errores de comunicaciones o de cualquier otro tipo en las peticiones modbus, la placa SiAM CRIO normalmente no responde nada, aunque en ciertos casos determinados puede responder con tramas de excepción.

La placa SiAM CRIO soporta tramas de broadcast dirigidas a la dirección 0. Estas tramas nunca serán respondidas.

8.2.- Mapeado Modbus

La placa SiAM CRIO mapea todas sus entradas, salidas y bits de estado interno en registros modbus accesibles, de acuerdo con la siguiente tabla.

Registros Modbus	Descripción	Tipo de acceso
0x0000-0x00FF	Entradas de memoria (%IW0…%IW255)	Sólo lectura
0x0100-0x01FF	Salidas de memoria (%QW0…%QW255)	Lectura/Escritura
0x0200-0x02FF	Memoria de sistema (%SW0…%SW255)	Lectura/Escritura
0x0300-0x03FF	Área reservada	Lectura/Escritura
0x1000-0x1FFF	Registros Modbus de direccionamiento indirecto	Lectura/Escritura
0x2000-0x2FFF	Registros Modbus configuración del direccionamiento indirecto.	Lectura/Escritura
0x4000-0x4FFF	RAM de usuario (%MW0…%MW255)	Lectura/Escritura
0x5000-0x5FFF	EEPROM de usuario (%VW0…%VW255)	Lectura/Escritura
0x6000-0x6FFF	Memoria de sistema (%SW0…%SW255)	Lectura/Escritura
0x7000-0x7FFF	Configuración	Sólo lectura
0x8000-0xFFFF	Programa de usuario	Sólo lectura

Sólo algunos de los 65536 registros disponibles se utilizan. Los otros son registros sólo de lectura y contienen el valor 0.

La zona de memoria de **Configuración** (0x7000-0x7FFF) y de **Programa** (0x8000-0xFFFF) sólo pueden ser leídas por el usuario. El software de configuración SiAM CRIOSoft posee también acceso de escritura a estas zonas.

La memoria de sistema esta mapeada en dos áreas diferentes del registro (0x0200-0x02FF and 0x6000-0x6FFFF). Las áreas de entradas, salidas, sistema, RAM y EPROM están descritas en detalle en la sección 7.4. Las áreas de direcciones indirectas – descritas previamente en la sección 6.8.2- y su configuración están descritas en los siguientes puntos.

8.2.1.- Registros de direccionemiento indirecto

Registro	Descripción
0x1000…0x103F	Registros de direccionamiento indirecto no volátil. Estos registros virtuales son redireccionados a los registros especificados en el rango 0x2000-0x201F tal y como se ha explicado anteriormente. Así pues, si el registro 0x2004 contiene el valor 10, acceder el registro 0x1004 equivaldrá a acceder al registro 10 (0x000A).
0x1100…0x113F	Registros de direccionamiento indirecto volátil. Estos registros virtuales son redireccionados a los registros especificados en el rango 0x2100-0x211F. Así pues, si el registro 0x2104 contiene el valor 10, acceder el registro 0x1104 equivaldrá a acceder al registro 10 (0x000A).
0x2000…0x203F	Configuración del direccionamiento indirecto no volátil. Estos registros contienen el número del registro al que se mapean los registros virtuales 0x1000-0x101F. Se almacenan en memoria no volátil y por tanto mantienen su valor tras una pérdida de alimentación.
0x2100…0x213F	Configuración del direccionamiento indirecto volátil. Estos registros contienen el número del registro al que se mapean los registros virtuales 0x1100-0x111F. Se almacenan en memoria volátil y por tanto se perderá su valor tras una pérdida de alimentación (al arrancar la placa todos los registros virtuales 0x1100-0x111F apuntaran al registro físico 0x0000).

8.2.2.- Registros de configuración

Registro	Descripción
0x7000	Identificador de dispositivo. Para este modelo siempre tiene el valor de 7001 (0x1B59)
0x7001	Versión de hardware
0x7002	Versión de firmware
0x7003…0x700F	String identificador del dispositivo. Estos registros contienen caracteres ASCII que identifican el dispositivo en la red modbus para el software SiAM CRIOSoft.
0x7010…0x7019	Reservado.
0x701A…0x7023	Valores mínimos para el escalado lineal de las entradas analógicas AI0…AI7 (lectura/escritura). Ver sección 6.8.3.
0x7024…0x702D	Valores máximos para el escalado lineal de las entradas analógicas AI0…AI7 (lectura/escritura). Ver sección 6.8.3.
0x702E	Modbus link timeout en milisegundos (lectura/escritura). Ver sección 6.8.1.



8.3.- Funciones modbus

La placa SiAM CRIO soporta las funciones del protocolo modbus RTU descritas –junto con ejemplos- en los apartados siguientes. Se debe tener en cuenta que las tramas están limitadas a 256 bytes, por lo que no es posible ni conveniente leer o escribir más de un centenar de registros modbus en una misma transacción. En todos los ejemplos se toma la dirección del esclavo modbus como 128 (0x80).

8.3.1.- Read Holding Registers (0x03)

Esta función permite leer un conjunto de registros modbus. En el siguiente ejemplo se lee el registro 0x0000:

Petición		Respuesta	
Campo	Valor (Hex)	Campo	Valor (Hex.)
Dirección de esclavo	0x80	Dirección de esclavo	0x80
Función	0x03	Función	0x03
Primer registro (HI)	0x00	Contador de bytes	0x02
Primer registro (LO)	0x00	Valor de Registro 0x0000 (HI)	0x00
Número de Registros (HI)	0x00	Valor de Registro 0x0000 (LO)	0x00
Número de Registros (LO)	0x01	Checksum (HI)	0x84
Checksum (HI)	0x9A	Checksum (LO)	0x5A
Checksum (LO)	0x1B		

8.3.2.- Read Input Registers (0x04)

Esta función es idéntica a la anterior. En el siguiente ejemplo se leen los registros 0x0200-0x0201.

Petición		Respuesta	
Campo	Valor (Hex.)	Campo	Valor (Hex.)
Dirección de esclavo	0x80	Dirección de esclavo	0x80
Función	0x04	Función	0x04
Primer registro (HI)	0x02	Contador de bytes	0x04
Primer registro (LO)	0x00	Valor de Registro 0x0200 (HI)	0x45
Número de Registros (HI)	0x00	Valor de Registro 0x0200 (LO)	0x02
Número de Registros (LO)	0x02	Valor de Registro 0x0201 (HI)	0x00
Checksum (HI)	0x6E	Valor de Registro 0x0201 (LO)	0x00
Checksum (LO)	0x62	Checksum (HI)	0xDE
		Checksum (LO)	0x40

El valor del registro 0x0200 es 0x4502, y el valor del registro 0x0201 es 0x0000.

8.3.3.- Write Single Register (0x06)

Esta función permite escribir un único registro modbus. En el siguiente ejemplo se escribe el valor 0xFFFF en el registro 0x0100:

Petición		Respuesta	
Campo	Valor (Hex.)	Campo	Valor (Hex.)
Dirección de esclavo	0x80	Dirección de esclavo	0x80
Función	0x06	Función	0x06
Dirección de registro(HI)	0x01	Dirección de registro (HI)	0x01
Dirección de registro (LO)	0x00	Dirección de registro (LO)	0x00
Valor del registro (HI)	0xFF	Valor del registro (HI)	0xFF
Valor del registro (LO)	0xFF	Valor del registro (LO)	0xFF
Checksum (HI)	0x9A	Checksum (HI)	0x9A
Checksum (LO)	0x1B	Checksum (LO)	0x1B

8.3.4.- Write Multple Registers (0x10)

Esta función permite escribir un conjunto registros modbus. El siguiente ejemplo, como el anterior, escribe el valor 0xFFFF en el registro 0x0100:

Petición		Respuesta	
Campo	Valor (Hex.)	Campo	Valor (Hex.)
Dirección de esclavo	0x80	Dirección de esclavo	0x80
Función	0x10	Función	0x10
Primer registro (HI)	0x01	Primer registro (HI)	0x01
Primer registro (LO)	0x00	Primer registro (LO)	0x00
Cantidad de registros (HI)	0x00	Cantidad de registros (HI)	0x00
Cantidad de registros (LO)	0x01	Cantidad de registros (LO)	0x01
Contador de bytes	0x02	Checksum (HI)	0x1E
Valor del registro 0x0100 (HI)	0xFF	Checksum (LO)	0x24
Valor del registro 0x0100 (LO)	0xFF		
Checksum (HI)	0xDB		
Checksum (LO)	0x76		

8.3.5.- Mask Write Register (0x16)

Esta función permite modificar el contenido de un registro modbus mediante una máscara AND y una máscara OR. Los bits activos de la máscara AND desactivan los bits respectivos del registro, y los bits activos de la máscara OR los activan. Esta función es muy útil para activar y desactivar bits individuales de los registros.

```
Result = (Current_Contents AND AND_MASK) OR (OR_MASK AND (NOT AND_MASK))
```

Por eso, si OR_MASK vale 0, el resultado es simplemente la lógica AND de *current_contents* y AND_MASK. Si la AND_MASK vale 0, el resultado es el mismo que el valor OR_MASK.

El siguiente ejemplo activa los bits 5 y 6 del Registro 0x0300:

Petición		Respuesta	
Campo	Valor (Hex.)	Campo	Valor (Hex.)
Dirección de esclavo	0x80	Dirección de esclavo	0x80
Función	0x16	Función	0x16
Dirección de registro (HI)	0x03	Dirección de registro (HI)	0x03
Dirección de registro (LO)	0x00	Dirección de registro (LO)	0x00
Mascara AND(HI)	0x00	Mascara AND (HI)	0x00
Mascara AND (LO)	0x00	Mascara AND (LO)	0x00
Mascara OR (HI)	0x00	Mascara OR (HI)	0x00
Mascara OR (LO)	0x60	Mascara OR (LO)	0x60
Checksum (HI)	0x3F	Checksum (HI)	0x3F
Checksum (LO)	0xB1	Checksum (LO)	0xB1

8.3.6.- Read / Write Multiple Registers (0x17)

Esta función permite leer y escribir conjuntos de registros en una única transacción modbus. Es la función ideal para leer entradas y escribir salidas de forma muy rápida, tal y como muestra el ejemplo siguiente:

Petición		**Respuesta**	
Campo	Valor (Hex.)	Campo	Valor (Hex.)
Dirección de esclavo	0x80	Dirección de esclavo	0x80
Función	0x17	Función	0x17
Primer registro a leer (HI)	0x00	Contador de bytes	0x02
Primer registro a leer (LO)	0x00	Valor del registro 0x0000 (HI)	0x00
Número de registros a leer (HI)	0x00	Valor del registro 0x0000 (LO)	0x00
Número de registros a leer (LO)	0x01	Checksum (HI)	0x81
Primer registro a escribir (HI)	0x01	Checksum (LO)	0xAA
Primer registro a escribir (LO)	0x00		
Número de registros a escribir (HI)	0x00		
Número de registros a escribir (LO)	0x01		
Contador de bytes	0x02		
Valor del registro 0x0100 (HI)	0xFF		
Valor del registro 0x0100 (LO)	0xFF		
Checksum (HI)	0xB8		
Checksum (LO)	0x9E		

9.- PREGUNTAS FRECUENTES

9.1.- No se puede establecer comunicación con la placa

Si no se puede establecer comunicación, deben comprobarse los siguientes puntos:

1. Revisar el cableado del bus RS-485 en la placa SiAM CRIO y en todos los demás elementos del bus (otras placas, conversores 232/485, etc.)
2. Comprobar que los jumpers de configuración de comunicaciones de la placa están ajustados correctamente:
    1. La resistencia de final de bus debe estar activa en el primer y el último dispositivo del bus RS-485.
    2. Las resistencias de pull-up y pull-down deben estar activadas solamente en uno de los dispositivos conectados al bus.
3. Si la placa sigue sin comunicar, observar los LEDs de diagnóstico de las comunicaciones.
    1. Si el LED de recepción (RX) no se ilumina, entonces la placa no está recibiendo datos por el bus. En ese caso puede haber un problema del conexionado o de que el maestro modbus no esté enviando datos (porque esté mal conectado o porque esté detenido).
    2. Si el LED de recepción se ilumina pero el LED de transmisión (TX) no lo hace, entonces la placa SiAM CRIO no está respondiendo a las peticiones modbus. Una causa de esto puede que las peticiones no estén dirigidas al esclavo modbus correcto. Revisar en ese caso la dirección del esclavo con la que intenta comunicar el maestro de la red.
4. Si todo está correcto, entonces el fallo de comunicación puede deberse a un problema hardware. Si el LED de estado de alimentación de la placa no parpadea, seguramente el microcontrolador de la placa estará dañado. Si parpadea pero aún así no es posible la comunicación, puede que algún componente de comunicaciones haya dejado de funcionar por alguna sobretensión en el bus de comunicaciones.

9.2.- Todas las salidas digitales están inactivas

Chequear el estado de las salidas online mediante el software de monitorización. Si las salidas no están activas en el software, hay que asegurarse que el programa las está activando. Si se están activando pero no activa su correspondiente led, hay que chequear los terminales de alimentación para ver si se está alimentando (9-30VDC). Ver sección 4.3 para una descripción detallada de los conectores de salida.

9.3.- El código IEC no se ejecuta o se ejecuta incorrectamente

Si las instrucciones no se ejecutan de la forma esperada, probablemente sea debido a que el acumulador lógico RLO tiene el valor FALSE, lo cual inhabilita la ejecución de muchas instrucciones por ejemplo, todas las instrucciones condicionales que terminan en 'C' como JMPC, CALC, RETC…). Por ello, tras acabar un segmento lógico (típicamente con una instrucción de almacenamiento ST o STN) e iniciar uno nuevo, es conveniente restablecer el valor del acumulador lógico RLO a TRUE. Esto puede hacerse mediante una instrucción LD TRUE, o mejor con una instrucción S (SET sin operando), que solamente ocupa 1 word de memoria de programa.



10.- CARACTERÍSTICAS ELÉCTRICAS

Parámetros	Min.	Tipo	Max.	Unidades
Temperatura de funcionamiento	0	-	70	ºC
Alimentación				
Tensión de alimentación continua V_{IN}	9	24	30	V
Corriente consumida	15[10]	-	110[11]	mA
Potencia consumida - V_{IN} = 9 VDC	0,1	-	1	W
Potencia consumida - V_{IN} = 12 VDC	0,2	-	1,3	W
Potencia consumida - V_{IN} = 24 VDC	0,4	-	2,6	W
Entradas digitales				
Impedancia de entrada	-	3,9	-	KΩ
Consumo de corriente por entrada - V_{IN} = 9VDC	-	-	2,3	mA
Consumo de corriente por entrada - V_{IN} = 12VDC	-	-	3,1	mA
Consumo de corriente por entrada - V_{IN} = 24VDC	-	-	6,2	mA
Entradas analógicas				
Impedancia de entrada – Modo corriente (0-20mA)	-	250	-	Ω
Salidas digitales				
Tensión de alimentación de salidas $V_{OUTPUTS}$	9	24	30	V
Máxima corriente en cada salida	-	-	80	mA
Programa				
Memoria de programa	8192 bytes (> 1300 IEC instrucciones)			
Memoria de usuario volátil (RAM)	256 words			
Memoria de usuario no volátil (EEPROM)	256 words			
Duración de la EEPROM	100K	1M	-	Writes
Niveles de anidamiento de paréntesis	4			
Niveles de anidamiento de subrutinas	4			
Tiempo de ejecución de instrucción IEC[12]	-	75	150	µs

[10] Sin ninguna salida activada
[11] Con todas las salidas activadas
[12] Depende del tipo de instrucción y sobretodo de si la monitorización de programa está activa.

11.- COPYRIGHT



Software i Automatismes Morvedre, S. L.
Vent de Ponent, 22
46520 Puerto de Sagunto
Valencia (Spain)
www.siam.es
info@siam.es